Advances in Intelligent Systems and Computing 655

Leszek Borzemski
Jerzy Świątek
Zofia Wilimowska *Editors*

Information Systems Architecture and Technology: Proceedings of 38th International Conference on Information Systems Architecture and Technology – ISAT 2017

Part I

Springer

Advances in Intelligent Systems and Computing

Volume 655

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.

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

Leszek Borzemski · Jerzy Świątek
Zofia Wilimowska
Editors

Information Systems Architecture and Technology: Proceedings of 38th International Conference on Information Systems Architecture and Technology – ISAT 2017

Part I

Springer

Editors

Leszek Borzemski
Department of Computer Science, Faculty of Computer Science and Management
Wrocław University of Science and Technology
Wrocław
Poland

Zofia Wilimowska
Department of Management Systems, Faculty of Computer Science and Management
Wrocław University of Science and Technology
Wrocław
Poland

Jerzy Świątek
Department of Computer Science, Faculty of Computer Science and Management
Wrocław University of Science and Technology
Wrocław
Poland

ISSN 2194-5357 ISSN 2194-5365 (electronic)
Advances in Intelligent Systems and Computing
ISBN 978-3-319-67219-9 ISBN 978-3-319-67220-5 (eBook)
DOI 10.1007/978-3-319-67220-5

Library of Congress Control Number: 2017952321

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

This three volume set of books includes the proceedings of the 2017 38th International Conference on Information Systems Architecture and Technology (ISAT), or ISAT 2017 for short, which will be held on September 17–19, 2017 in Szklarska Poręba, Poland. The conference was organized by the Department of Computer Science and Department of Management Systems, Faculty of Computer Science and Management, Wrocław University of Technology, Poland.

The International Conference on Information Systems Architecture has been organized by the Wrocław University of Technology from the seventies of the last century. The purpose of the ISAT is to discuss a state of the art of information systems concepts and applications as well as architectures and technologies supporting contemporary information systems. The aim is also to consider an impact of knowledge, information, computing and communication technologies on managing the organization scope of functionality as well as on enterprise information systems design, implementation, and maintenance processes taking into account various methodological, technological, and technical aspects. It is also devoted to information systems concepts and applications supporting exchange of goods and services by using different business models and exploiting opportunities offered by Internet-based electronic business and commerce solutions.

ISAT is a forum for specific disciplinary research, as well as for multi-disciplinary studies to present original contributions and to discuss different subjects of today's information systems planning, designing, development, and implementation. The event is addressed to the scientific community, people involved in variety of topics related to information, management, computer and communication systems, and people involved in the development of business information systems and business computer applications.

This year, we received 180 papers. The papers included in the three proceedings volumes published by Springer have been subject to a thoroughgoing review process by highly qualified peer reviewers. At least two members of Program Committee or Board of Reviewers reviewed each paper. The final acceptance rate was 56%. Program Chairs selected 101 best papers for oral presentation and publication in the 38th International Conference on Information Systems Architecture and Technology 2017 proceedings.

The papers have been grouped into three volumes:

Part I—discoursing about topics including, but not limited to, Artificial Intelligence Methods, Knowledge Discovery and Data Mining, Big Data, Knowledge Discovery and Data Mining, Knowledge Based Management, Internet of Things, Cloud Computing and High Performance Computing, Distributed Computer Systems, Content Delivery Networks, Service Oriented Computing, and E-Business Systems, Web Design, Mobile and Multimedia Systems.

Part II—addressing topics including, but not limited to, System Modelling for Control, Recognition and Decision Support, Mathematical Modeling in Computer System Design, Service Oriented Systems and Cloud Computing and Complex process modelling.

Part III—dealing with topics including, but not limited to, Modeling of Manufacturing Processes, Modeling an Investment Decision Process, Management of Innovation, Management of Organization.

We would like to thank the Program Committee and external reviewers, essential for reviewing the papers to ensure a high standard of the ISAT 2017 conference and the proceedings. We thank the authors, presenters, and participants of ISAT 2017; without them, the conference could not have taken place. Finally, we thank the organizing team for the efforts in bringing the conference to a successful scientific event.

We devote ISAT 2017 to our friend and former ISAT Chair, Professor Adam Grzech.

September 2017

Leszek Borzemski
Jerzy Świątek
Zofia Wilimowska

In Memory of Our Friend and Past ISAT Chair

Professor Adam Grzech

November 2016

Professor DSc. Adam Grzech was born in 1954 in Dębica (Poland). He was graduated at the Wroclaw University of Technology 1977—M.Sc. Electronic Engineering. He did his PhD at the Institute of Technical Cybernetics in 1979 and D.Sc. in technical sciences in the field of computer science in 1989 and received the title of full Professor in 2003 from Wroclaw University of Technology.

He was an Assistant Professor at the Institute of Technical Cybernetics (1979–1982), an Assistant Professor at the Institute of Control and Systems Engineering (1982–1989), Associate Professor at the Institute of Control and Systems Engineering (1989–1993), a Professor at the Institute for Technical Informatics (1993–2006), and a Full Professor at the Institute of Computer Science, Faculty of Computer Science and Management at Wroclaw University of Technology (since 2006).

He was an author and co-author of over 350 published research works. His areas of research were as follows: analysis, modeling, and design information and communication systems and networks. Since 2002, he was a Chief of Telecommunication Department. He was a leader of Scientific School on Information and Communication Systems and Networks. He was a promoter of the 10 completed Ph.D. theses.

During the years 1991–1993 and 1999–2002, he was a Director of the Institute of Control and Systems Engineering. Since 2002, he was a delegate of the Rector for Information Technology, and then during the years 2003–2005, he was a Vice President for Development at the Wroclaw University of Technology and in 2002–2008 a member of the Senate of the Wroclaw University of Technology. During the years 2005–2012, he was a Vice Dean of Computer Science and Management Faculty.

He was active in the work at the national, international committees of conferences and scientific journals. He was Co-chair of Information Systems Architecture and Technology (ISAT) and Systems Science International Conferences. Since 1982, he served as Scientific Secretary, and since 2006 Editor-in-Chief of the Science Systems journal published quarterly.

Professor Adam Grzech was a member of the Wroclaw Scientific Society, the Polish Informatics Society, the Council of Information Technology, the Committee on Informatics of the Polish Academy of Sciences, and Technical Committee TC6 (Communication Systems) IFIP.

Leszek Borzemski
Jerzy Świątek
Zofia Wilimowska

ISAT 2017 Szklarska Poręba, September 17–19, 2017

ISAT 2017 Conference Organization

General Chair

Leszek Borzemski, Poland

Program Co-chairs

Leszek Borzemski, Poland
Jerzy Świątek, Poland
Zofia Wilimowska, Poland

Local Organizing Committee

Leszek Borzemski (Chair)
Zofia Wilimowska (Co-chair)
Jerzy Świątek (Co-chair)
Mariusz Fraś (Conference Secretary, Website Support)
Arkadiusz Górski (Technical Editor)
Anna Kamińska (Technical Secretary)
Ziemowit Nowak (Technical Support)
Kamil Nowak (Website Coordinator)
Danuta Seretna-Sałamaj (Technical Secretary)

International Program Committee

Leszek Borzemski, Poland (Chair)
Jerzy Świątek, Poland (Co-chair)
Zofia Wilimowska, Poland (Co-chair)
Witold Abramowicz, Poland

Dhiya Al-Jumeily, UK
Iosif Androulidakis, Greece
Patricia Anthony, New Zealand
Zbigniew Banaszak, Poland
Elena N. Benderskaya, Russia
Janos Botzheim, Japan
Djallel E. Boubiche, Algeria
Patrice Boursier, France
Anna Burduk, Poland
Andrii Buriachenko, Ukraine
Udo Buscher, Germany
Wojciech Cellary, Poland
Haruna Chiroma, Malaysia
Edward Chlebus, Poland
Gloria Cerasela Crisan, Romania
Marilia Curado, Portugal
Czesław Daniłowicz, Poland
Zhaohong Deng, China
Małgorzata Dolińska, Poland
Milan Edl, Czech Republic
El-Sayed M. El-Alfy, Saudi Arabia
Peter Frankovsky, Slovakia
Naoki Fukuta, Japan
Bogdan Gabryś, Poland
Piotr Gawkowski, Poland
Manuel Graña, Spain
Wiesław M. Grudzewski, Poland
Katsuhiro Honda, Japan
Marian Hopej, Poland
Zbigniew Huzar, Poland
Natthakan Iam-On, Thailand
Biju Issac, UK
Arun Iyengar, USA
Jürgen Jasperneite, Germany
Janusz Kacprzyk, Poland
Henryk Kaproń, Poland
Yury Y. Korolev, Belarus
Yannis L. Karnavas, Greece
Ryszard Knosala, Poland
Zdzisław Kowalczuk, Poland
Lumír Kulhanek, Czech Republic
Binod Kumar, India
Jan Kwiatkowski, Poland
Antonio Latorre, Spain
Radim Lenort, Czech Republic

Gang Li, Australia
José M. Merigó Lindahl, Chile
Jose M. Luna, Spain
Emilio Luque, Spain
Sofian Maabout, France
Lech Madeyski, Poland
Zygmunt Mazur, Poland
Elżbieta Mączyńska, Poland
Pedro Medeiros, Portugal
Toshiro Minami, Japan
Marian Molasy, Poland
Zbigniew Nahorski, Poland
Kazumi Nakamatsu, Japan
Peter Nielsen, Denmark
Tadashi Nomoto, Japan
Cezary Orłowski, Poland
Sandeep Pachpande, India
Michele Pagano, Italy
George A. Papakostas, Greece
Zdzisław Papir, Poland
Marek Pawlak, Poland
Jan Platoš, Czech Republic
Tomasz Popławski, Poland
Edward Radosinski, Poland
Wolfgang Renz, Germany
Dolores I. Rexachs, Spain
José S. Reyes, Spain
Leszek Rutkowski, Poland
Sebastian Saniuk, Poland
Joanna Santiago, Portugal
Habib Shah, Malaysia
J.N. Shah, India
Jeng Shyang, Taiwan
Anna Sikora, Spain
Marcin Sikorski, Poland
Małgorzata Sterna, Poland
Janusz Stokłosa, Poland
Remo Suppi, Spain
Edward Szczerbicki, Australia
Eugeniusz Toczyłowski, Poland
Elpida Tzafestas, Greece
José R. Villar, Spain
Bay Vo, Vietnam
Hongzhi Wang, China
Leon S.I. Wang, Taiwan

Junzo Watada, Japan
Eduardo A. Durazo Watanabe, India
Jan Werewka, Poland
Thomas Wielicki, USA
Bernd Wolfinger, Germany
Józef Woźniak, Poland
Roman Wyrzykowski, Poland
Yue Xiao-Guang, Hong Kong
Jaroslav Zendulka, Czech Republic
Bernard Ženko, Slovenia

ISAT 2017 Reviewers

Małgorzata Adamska, Poland
Patricia Anthony, New Zealand
Katarzyna Antosz, Poland
Zbigniew Antoni Banaszak, Poland
Jan Betta, Poland
Grzegorz Bocewicz, Poland
Leszek Borzemski, Poland
Janos Botzheim, Hungary
Djallel Eddine Boubiche, Algeria
Krzysztof Brzostowski, Poland
Anna Burduk, Poland
Udo Buscher, Germany
Wojciech Cellary, Poland
Haruna Chiroma, Malaysia
Witold Chmielarz, Poland
Grzegorz Chodak, Poland
Kazimierz Choroś, Poland
Piotr Chwastyk, Poland
Gloria Cerasela Crisan, Romania
Anna Czarnecka, Poland
Mariusz Czekała, Poland
Grzegorz Debita, Poland
Jarosław Drapała, Poland
Tadeusz Dudycz, Poland
Fic Maria, Poland
Mariusz Fraś, Poland
Naoki Fukuta, Japan
Joanna Furman, Poland
Piotr Gawkowski, Poland
Krzysztof Goczyła, Poland
Arkadiusz Gola, Poland

Arkadiusz Górski, Poland
Jerzy Grobelny, Poland
Joanna Helman, Poland
Bogumila Hnatkowska, Poland
Katsuhiro Honda, Japan
Grażyna Hołodnik-Janczura, Poland
Zbigniew Huzar, Poland
Artur Hłobaż, Poland
Biju Issac, UK
Katarzyna Jach, Poland
Małgorzata Jasiulewicz-Kaczmarek, Poland
Jerzy Józefczyk, Poland
Ireneusz Jóźwiak, Poland
Magdalena Jurczyk-Bunkowska, Poland
Krzysztof Juszczyszyn, Poland
Robert Kamiński, Poland
Yannis Karnavas, Greece
Włodzimierz Kasprzak, Poland
Grzegorz Kołaczek, Poland
Mariusz Kołosowski, Poland
Jacek Korniak, Poland
Zdzisław Kowalczuk, Poland
Damian Krenczyk, Poland
Lumír Kulhánek, Czech Republic
Binod Kumar, India
Jan Kwiatkowski, Poland
Sławomir Kłos, Poland
Antonio LaTorre, Spain
Zbigniew Lipiński, Poland
Wojciech Lorkiewicz, Poland
Andrzej Loska, Poland
Sofian Maabout, France
Lech Madeyski, Poland
Rafał Michalski, Poland
Miroslava Mlkva, Slovak Republic
Marian Molasy, Poland
Zbigniew Nahorski, Poland
Peter Nielsen, Denmark
Cezary Orłowski, Poland
Donat Orski, Poland
Michele Pagano, Italy
Agnieszka Parkitna, Poland
Iwona Pisz, Poland
Jan Platoš, Czech Republic
Grzegorz Popek, Poland

Tomasz Popławski, Poland
Sławomir Przyłucki, Poland
Dolores Rexachs, Spain
Maria Rosienkiewicz, Poland
Stefano Rovetta, Italy
Jacek, Piotr Rudnicki, Poland
Dariusz Rzońca, Poland
Anna Saniuk, Poland
Joanna Santiago, Portugal
José Santos, Spain
Danuta Seretna-Sałamaj, Poland
Anna Sikora, Spain
Marcin Sikorski, Poland
Dorota Stadnicka, Poland
Malgorzata Sterna, Poland
Janusz Stokłosa, Poland
Grażyna Suchacka, Poland
Remo Suppi, Spain
Joanna Szczepańska, Poland
Edward Szczerbicki, Poland
Jerzy Świątek, Poland
Błażej Święcicki, Poland
Kamila Urbańska, Poland
José R. Villar, Spain
Bay Vo, Vietnam
Anna Walaszek-Babiszewska, Poland
Leon Shyue-Liang Wang, Taiwan
Krzysztof Waśko, Poland
Jan Werewka, Poland
Zofia Wilimowska, Poland
Bernd Wolfinger, Germany
Józef Woźniak, Poland
Ryszard Wyczółkowski, Poland
Jacek Zabawa, Poland
Krzysztof Zatwarnicki, Poland
Jaroslav Zendulka, Czech Republic
Maciej Zięba, Poland

ISAT 2017 Special Session

Modeling of Manufacturing Processes
Anna Burduk (Chair), Poland

Contents

Artificial Intelligence Methods

A Deep Learning Approach for Valve Defect Recognition in Heart Acoustic Signal

Dariusz Kucharski, Dominik Grochala(✉), Marcin Kajor, and Eliasz Kańtoch

AGH University of Science and Technology, Al. Mickiewicza 30, 30-059 Kraków, Poland
{darekk,grochala,mkajor,kantoch}@agh.edu.pl

Abstract. The analysis of phonocardiogram (PCG), although considered as well established in a clinical application, still constitutes the valuable source of diagnostic data. Currently, electronic auscultation provides digital signals which can be processed in order to automatically evaluate the condition of heart or lungs. In this paper, we propose a novel approach for the classification of phonocardiographic signals. We extracted a set of time-frequency parameters which enable to effectively differentiate between normal and abnormal heart beats (with valve defects). These features have constituted an input of the convolutional neural network, which we used for classification of pathological signals. The Aalborg University heart sounds database from PhysioNet/Computing in Cardiology Challenge 2016 was used for verification of developed algorithms. We obtained 99.1% sensitivity and 91.6% specificity on the test data, which is motivational for further research.

Keywords: Deep learning · Heart sound classification · Convolutional neural network · Machine learning · Signal processing

1 Introduction

Heart diseases constitute one of the most widespread causes of death worldwide. More than 17 million people died because of cardiovascular issues in 2012, which corresponds to 31% of all global deaths [1]. The significant number of total cardiac disorders include valves defects – the cause of mitral stenosis (MS) and mitral regurgitation (MR). These diseases are especially common in industrially developed countries (1.8% of prevalence) [2]. Nowadays, advanced medical devices provide numerous, complex diagnostic methods. Nevertheless, an auscultation technique is still popular within physicians because it enables to quick examination, especially in the case of heart screening. This, in turn, leads to initial disease evaluation and constitutes the ground for further diagnostic procedures. Every mechanically active internal organ generates sounds of particular spectral characteristics, which is the base for this kind of examination. In every cardiac cycle, the heart muscle is electrically stimulated to induce atrial and ventricular contractions. This mechanical activity constitutes a driving force for blood flow in the heart and the whole cardiovascular system. Sudden flux disturbances, caused by the blood being obstructed by internal heart walls and valves, generate vibrations of the

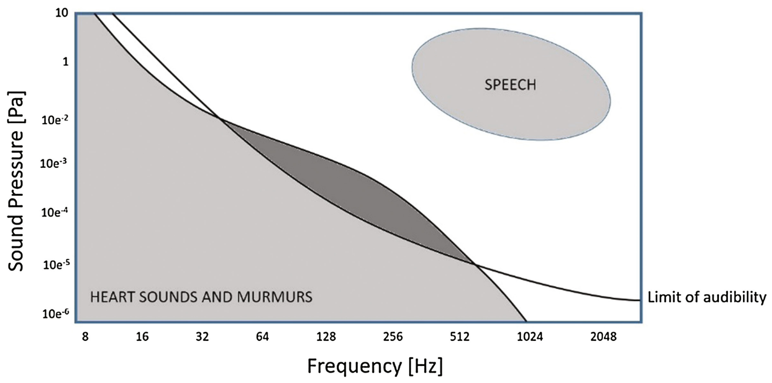


Fig. 1. The spectral characteristic of physiological heart sounds murmurs and speech during cardiac auscultation [1].

entire cardiac structure. These acoustic signals are audible on the chest surface and can be utilized for evaluation of heart condition. Four locations in which the cardiac sounds can be recorded most effectively are mitral, tricuspid, pulmonic and aortic areas. The cardiac acoustic signals are usually classified into a set of characteristic tones. The first one (S1 - systolic) is a result of the sudden closure of mitral and tricuspid valves. This sound consists of two components, generated by each valve, however, in physiological conditions, they are practically indistinguishable. The second tone (S2 – diastolic) occurs when the aortic and pulmonic valves are closed – physiologically, the louder aortic component is emitted before the pulmonic one. Apart from S1 and S2 tones, which are considered as fundamental, the other characteristic sounds such as S3, S4 or systolic ejection (MC) may occur during the cardiac cycle. In case of heart disorder, turbulent blood flow generates irregular sounds of wide spectrum called murmurs [1].

Currently, electronic medical systems provide digital data recording which enables to use computers for advanced signal processing. This leads to increasing the sound quality and more often - automatic diagnosis Nevertheless, spectral analysis of cardiac acoustics proves that both physiological and pathological signals, as well as respiration artifacts, overlap in frequency domain (Table 1). This is one of the major aspects which makes the automatic classification of heart sounds so difficult. The Fig. 1 represents spectral regions of different heart sounds in comparison with speech and limit of audibility.

Table 1. The comparison of spectral energy range of sounds occurring during auscultation [1].

Sound	Typical frequency range	Comment
S1	10–140 Hz	Highest energies in range 25–40 Hz
S2	10–200 Hz	Highest energies in range 55–75 Hz
S3 and S4	20–70 Hz	None
Murmurs	Up to 600 Hz	None
Respiration	200–700 Hz	Highest energies in range 25–40 Hz

An automated analysis of cardiac acoustic signals in clinical applications is usually based on three main steps: preprocessing, segmentation and classification (Fig. 2). The first one aims to filter artifacts but also assess the quality and extract the representative features of the sound. In the second step, the signal is segmented into the specific phases of heart cyclic activity – this constitutes a base for the third step which is classification.

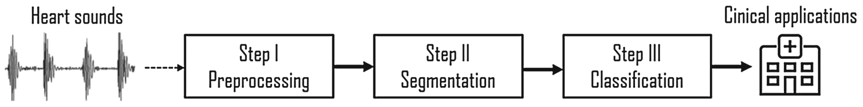


Fig. 2. Fundamental steps in the automatic analysis of heart acoustic signals.

Recently, many approaches have been introduced into segmentation of heart sounds. The most common methods include envelope-based algorithms, feature-based methods, machine learning and Hidden Markov Model (HMM). Sun et al. stated the 97% accuracy of cardiac acoustic signal segmentation with the use of short-time modified Hilbert Transform for envelope calculation. The method was evaluated on 7730 s of abnormal and 600 s of normal PCG and 1496.8 s from Michigan MHSDB database [3]. Varghees and Ramachandran reported that Shannon entropy in conjunction with instantaneous phase feature calculated from the analytical signal can be used for perform effective heart sound segmentation. They achieved an average sensitivity of 99.43% and positive predictivity of 93.56% without splitting a data into training and testing sets, using 701 segments of both noisy and clean and physiological and pathological signals [4]. Tang et al. developed a dynamic clustering algorithm based on time-frequency parameters of instantaneous cycle frequency. With this method, they achieved an accuracy of 94.9% for S1 and 95.9% for S2 tested on 25 subjects [5]. Sedighian et al. used the HMM with homomorphic filtering and obtained mean accuracy of 92.4% for S1 and 93.5% for S2 segmentation on PASCAL database [6].

The automated classification of pathological heart acoustic signals usually can be divided into basic groups of algorithms and of most of them are based on machine learning approach. Among them, the Artificial Neural Networks (ANN) and Support Machine Vector (SMV) classification are the most popular. However, the use of HMM-based and clustering-based classification were also widely reported in terms of heart sound analysis [1]. Uguz described the methodology involving time-frequency features as input for the ANN for classification of normal, pulmonary and mitral stenosis heart valve diseases. In this research, the total of 120 sounds was split 50/50 into train and test set. Reported results included 90.48% sensitivity and 95% accuracy [7]. In contrary, Zheng et al. stated that coefficients of wavelet packets decomposition of heart sounds and sample entropy can be successfully used as input for SMV classifier. The method was tested on 40 normal and 67 pathological signals and enabled to obtain 97.17% accuracy and 93.48% sensitivity [8].

2 Methods

Recently, it has been stated that neural networks are experiencing the Renaissance, because since 2012 machine learning has been gaining more and more attention all over the scientific world. It has started with ILSVRC 2012 where AlexNet model outclassed other competitors. It scored top-5 test error rate of 15.3% compared to 26.2% achieved in second place [9]. AlexNet used a neural network with convolutional architecture (CNN). From that day on, a great progress has been made in the field of CNN [10] not only in vision systems but also in other technical branches like signal processing or speech recognition [11]. In comparison with traditional neural networks, the convolutional classifier has its advantage mainly in automated feature extraction part of the entire algorithm. The first part of this model is usually used for representative parameters selection. It generally consists of convolutional and decimation layers (like maxpooling) and classification part, which is usually based on MLP and softmax layers as an output. As in our research, we took advantage of spectrogram as a feature vector, we decided to adapt an image-like recognition approach, as the research of using the spectrogram as network input has already been conducted in regard to acoustic applications [11].

During our research, we noticed that heart beat with valve defect features spectral component around 400–800 Hz which is in opposition to normal cardiac sound, where in high frequencies only noisy components can be found. This result was a prelude for spectrogram analysis of segmented heart sound. We took 8 s of signal and calculate spectrogram with 500-samples window and 450-samples overlap. We noticed the coefficient stripes similar to single beat spectrum. This observation was a motivation for using spectrograms as an input for the convolutional neural network. One advantage, resulting from calculating the spectrogram for signal sections with a particular amount of normal and abnormal segments, is the possibility to avoid the obligatory heart beat

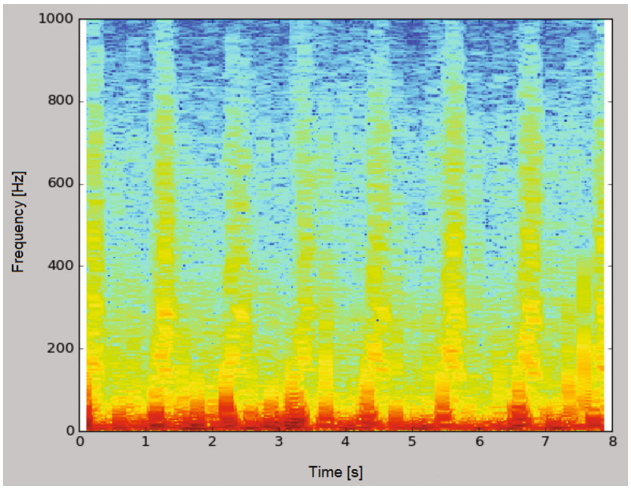


Fig. 3. An example of spectrogram for the pathological signal. Heart beats are visible almost at all frequencies

detection as separated algorithm step. Moreover, we assume that convolutional layers in neural network can be used for extract more significant features. Finally, there are some examples where spectrograms are successfully used as input e.g. for speech recognition [11]. Figures represent spectrograms of the pathological and physiological heart sounds (Figs. 3 and 4).

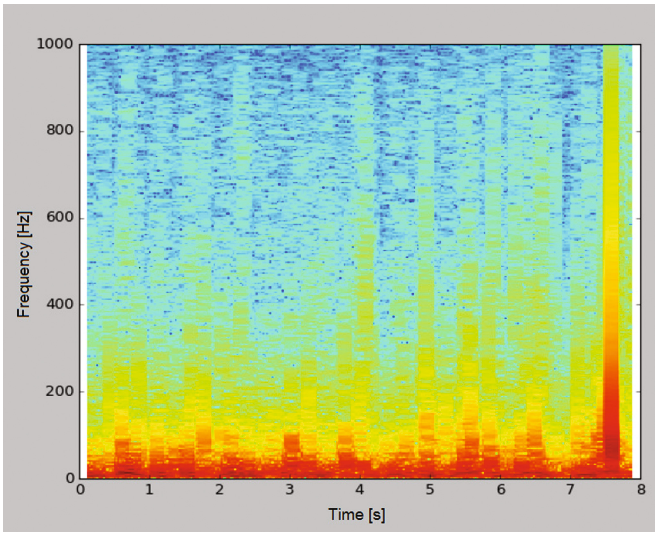


Fig. 4. An example of spectrogram for the physiological signal. Heart beats are visible at low frequencies.

2.1 Data Preparation

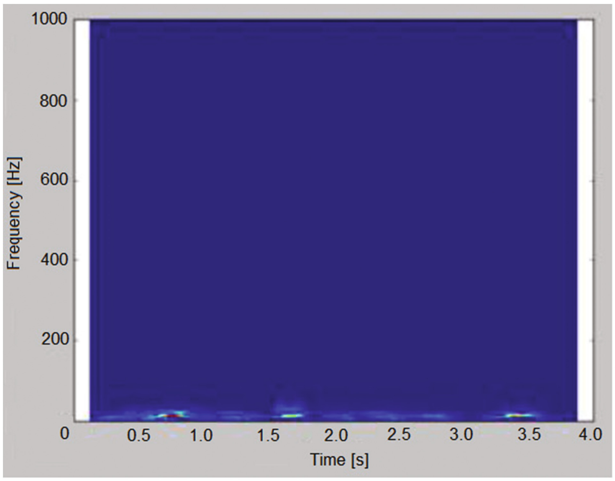
Physiological and pathological signals containing valve defects were taken from Aalborg University heart sounds database (PhysioNet/Computing in Cardiology Challenge 2016). 66 recordings of 8 s length (33 with a disorder and 33 with clean heart beats) was split into train set, validation set and test set with the proportion of 0.8: 0.1: 0.1. We split dataset at the very beginning of preprocessing in order to avoid overfitting and to receive reliable results. The characteristic of prepared datasets are listed below:

– train set: 984 examples of physiological signals and 923 examples of pathological
– validation set: 121 examples of physiological signals and 123 pathological
– test set: 123 examples of physiological signals and 123 examples of pathological

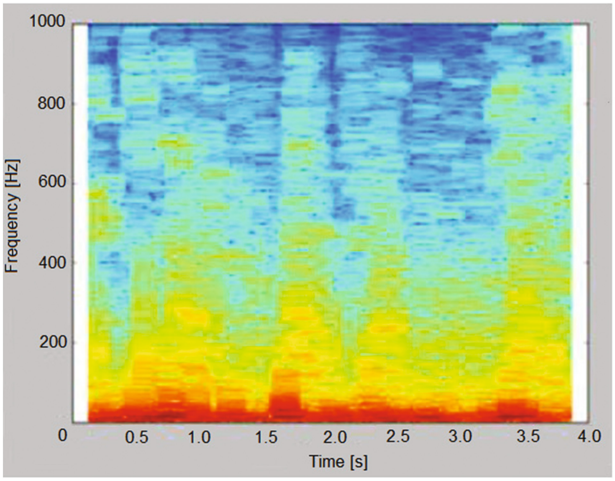
2.2 Preprocessing

For each signal segment, the spectrogram was calculated. We only have chosen amplitude value of Fourier Transform. In the next step, we calculated natural logarithm values on each spectrogram to reduce high values. Then, a simple normalization was applied.

We subtracted the minimum value from each sample and divided the result by maximum of the entire dataset. In the last step of preprocessing, we snipped spectrogram in frequency axis between 640 and 876 Hz, as we have found that most valuable information is contained around these frequencies. The (Figs. 5 and 6) depicts the steps performed during preprocessing stage.

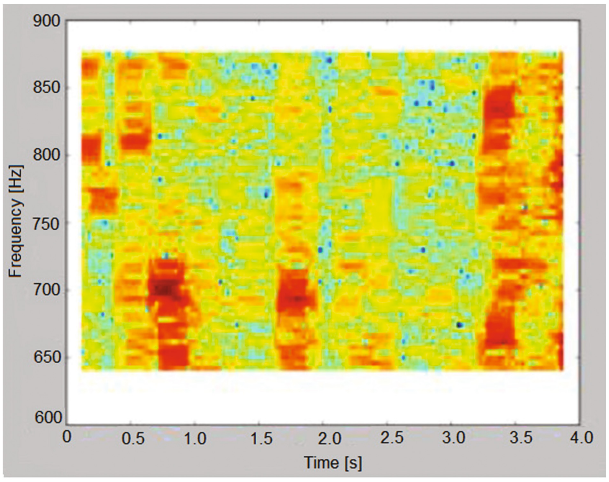


a)

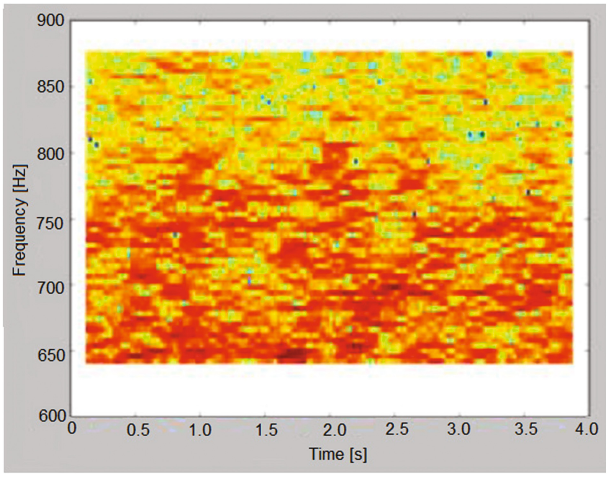


b)

Fig. 5. Preprocessing steps. (a) An example of absolute values of the spectrogram. (b) Spectrogram after normalization and calculation of logarithmic values.



a)



b)

Fig. 6. Further preprocessing steps (a) A part of spectrogram snipped out between 640 and 876 Hz (b) As contrast, a physiological example of spectrogram snipped between 640 and 876 Hz.

2.3 ANN Model Architecture

We used CNN with a fully connected layer on the end of convolutional part and softmax with two outputs – physiological and pathological. Input has a size of the 60×151 matrix with each cell value between 0 and 1. One cell represents 26.5 ms in the time axis and 3.93 Hz in the frequency domain. The convolutional part consists of three convolutional layers, each of which is followed by max pooling layer of window size 2×2. The convolutional layer has respectively: 16 kernels of size 7×7, 32 kernels of size 5×5 and 32 kernels of 3×3. After last max pooling layer, there is additional convolutional layer with 32 filters of size 3 × 3 and another convolutional layer with 16 filters of size 1 × 1, which was used to reduce the dimensionality of extracted features. The result from the convolutional part is 16 features map of size 1 × 7 (Fig. 7).

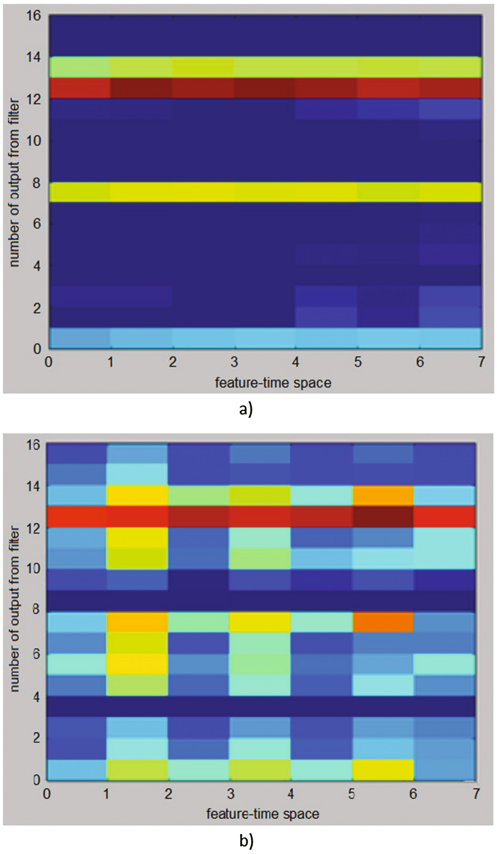


Fig. 7. An example of the convolutional part output. (a) A feature map for the physiological signal, (b) the pathological one.

The next layer which follows convolutional part is the classification layer. It consists of the fully connected layer with 32 neurons and softmax layer with two outputs. We also added dropout layer before both fully connected layers and softmax to prevent the network from overfitting [12]. Our tests showed that 50% dropout for the fully connected layer and 20% for softmax layer gave the best results. The developed CNN topology is presented in the (Fig. 8).

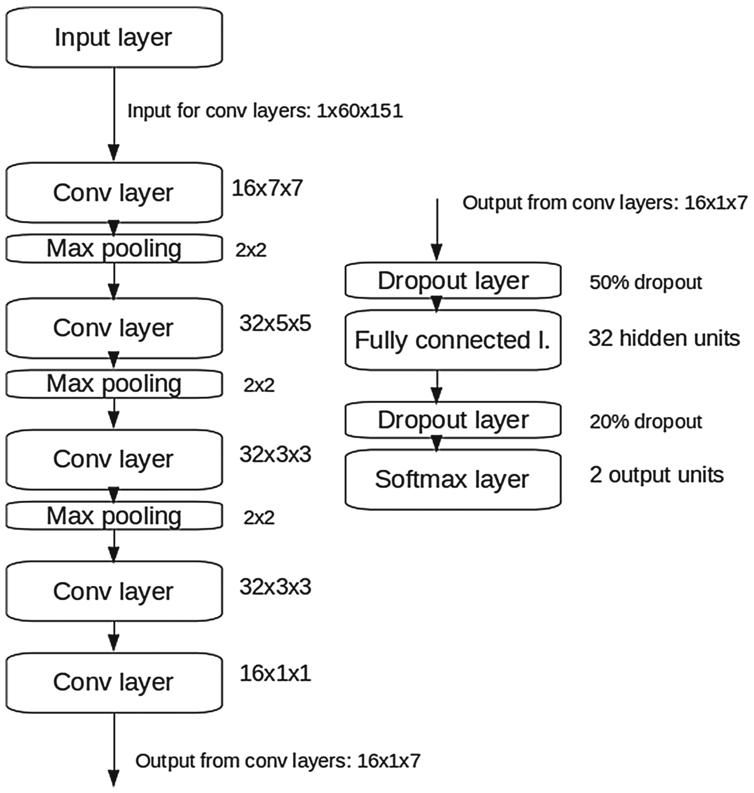


Fig. 8. CNN architecture.

2.4 ANN Training

We applied Adagrad optimization method [13] with 0.001 learning rate. According to validation set loss values, we stopped training at the moment when the network started to overfit (about 27000 epoch) (Fig. 9).

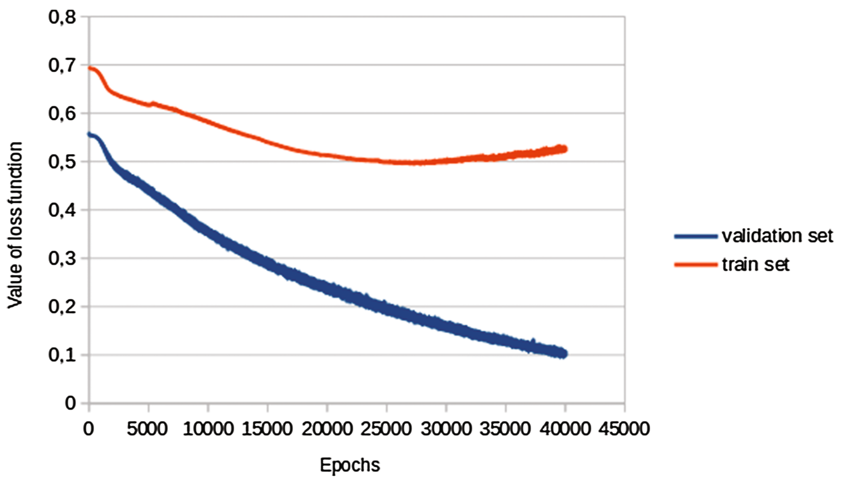


Fig. 9. Loss function for a train set and validation set. One can see an overfitting started at about 27000 epoch.

3 Result and Discussion

We noticed that for higher learning rate, the loss function for our model could not converge to a minimum. We also tested gradient descent with momentum (for learning rate <0.1;0.001> and momentum <0.5;0.9>) but this optimization method gave a worse final result (Fig. 10).

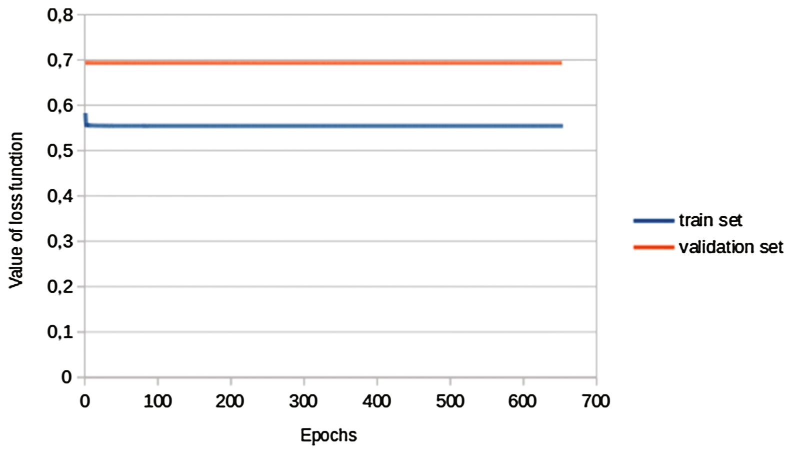


Fig. 10. An example for learning rate of 0.01. Too high learning rate caused optimization algorithm to miss minimum.

Finally, learning took 27000 epoch with 0.19 loss error on the training set and 0.49 on the validation set. The final result was calculated on the test set. We achieved sensitivity equal to 99,1% and specificity equal to 91,6%. Final results are presented in Tables 2 and 3.

Table 2. Confusion matrix

	Normal class detected by our model	Abnormal class detected by our model
Normal class	120	1
Abnormal class	11	112

Table 3. Statistics of our result.

	Sensitivity	Specificity
CNN	99.1%	91.6%

4 Conclusion

In this research, we evaluated the novel approach in heart sound classification. Developed algorithm is based on the convolutional neural network which uses the spectrogram as input feature vector. Obtained results are comparable with most of the state-of-the-art works. However, the use of deep learning methods in the analysis of heart acoustic signals is certainly a modern approach and the satisfactory classification score is motivational for further research.

The lack of well-documented, extensive heart sound databases contributes to the difficulties concerning signal quality and spectral variation within the gathered data. During research we took advantage of Aalborg University heart sounds database which constitutes the part of “An open access database for the evaluation of heart sound algorithms” (PhysioNet/Computing in Cardiology Challenge 2016) - virtually, the only cardiac sound database with the amount of signals sufficient for machine learning. This leads to the conclusion that development of tailor-made heart sound database with well defined, repeatable measurement conditions may significantly improve the quality of further work.

Automatic heart disorder recognition is an essential problem for screening examinations where thousands of patient need to be examined in a short period of time, therefore, development of such algorithms is a vital need for modern medicine.

References

1. Liu, C., Springer, D., Li, Q., Moody, B., Juan, R.A., Chorro, F.J., Castells, F., Roig, J.M., Silva, I., Johnson, A.E.W., Syed, Z., Schmidt, S.E., Papadaniil, C.D., Hadjileontiadis, L., Naseri, H., Moukadem, A., Dieterlen, A., Brandt, C., Tang, H., Samieinasab, M., Samieinasab, M.R., Sameni, R., Mark, R.G., Clifford, G.D.: An open access database for the evaluation of heart sound algorithms. Physiol. Meas. **37**, 2181–2213 (2016)

2. Ray, R., Chambers, J.: Mitral valve disease. Int. J. Clin. Pract. **68**, 1216–1220 (2014)
3. Sun, S., Jiang, Z., Wang, H., Fang, Y.: Automatic moment segmentation and peak detection analysis of heart sound pattern via short-time modified Hilbert transform. Comput. Methods Programs Biomed. **114**, 219–230 (2014)
4. Varghees, V.N., Ramachandran, K.I.: A novel heart sound activity detection framework for automated heart sound analysis. Biomed. Signal Process. Control **13**, 174–188 (2014)
5. Tang, H., Li, T., Qiu, T., Park, Y.: Segmentation of heart sounds based on dynamic clustering. Biomed. Signal Process. Control **7**, 509–516 (2012)
6. Sedighian, P., Subudhi, A.W., Scalzo, F., Asgari, S.: Pediatric heart sound segmentation using Hidden Markov Model. In: 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 5490–5493. IEEE (2014)
7. Uguz, H.: A biomedical system based on artificial neural network and principal component analysis for diagnosis of the heart valve diseases. J. Med. Syst. **36**, 61–72 (2012)
8. Zheng, Y., Guo, X., Ding, X.: A novel hybrid energy fraction and entropy-based approach for systolic heart murmurs identification. Expert Syst. Appl. **42**, 2710–2721 (2015)
9. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 25, pp. 1097–1105. Curran Associates, Inc. (2012)
10. Karpathy, A., Fei-Fei, L.: Deep visual-semantic alignments for generating image descriptions. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
11. Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., et al.: Deep speech 2: end-to-end speech recognition in english and mandarin. In: International Conference on Machine Learning, pp. 173–182 (2016)
12. Srivastava, N., Hinton, G., Krizhevsky, A.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Res. **15**, 1929–1958 (2014)
13. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. **12**, 2121–2159 (2011)

Population-Based Algorithm with Selectable Evolutionary Operators for Nonlinear Modeling

Krystian Łapa(✉)

Institute of Computational Intelligence,
Częstochowa University of Technology, Częstochowa, Poland
krystian.lapa@iisi.pcz.pl

Abstract. In this paper a new population-based algorithm for nonlinear modeling is proposed. Its advantage is the automatic selection of evolutionary operators and their parameters for individuals in population. In this approach evolutionary operators are selected from a large set of operators, however only the solutions that use low number of operators are promoted in population. Moreover, assigned operators can be changed during evolution of population. Such approach: (a) eliminates the need for determining detailed mechanism of the population-based algorithm, and (b) reduces the complexity of the algorithm. For the simulations typical nonlinear modeling benchmarks are used.

Keywords: Population-based algorithms · Fuzzy systems · Nonlinear modeling · Selection of evolutionary operators

1 Introduction

Nonlinear modeling issues can be solved using, among the others, population-based algorithms which are part of computational intelligence methods (see e.g. [4]). The idea of population-based algorithms relies on iterative processing of individuals. In each iteration a new generation of individuals is generated (characterized, by definition, by improved values of the objective function). Population-based algorithms differ from traditional optimization methods: (a) they do not process parameters directly, but their encoded form, (b) they search solutions based not on a single point, but on a population of the points (the population contains individuals, each individual encodes single solution), (c) they use objective function directly, not its derivatives (this function determines the quality of the solutions in the population), (d) they use probabilistic mechanisms, not deterministic. As a result, they have advantage over other optimization techniques such as: analytical methods, random methods, etc. [6].

Population-based algorithms can be also combined with other computational intelligence methods, such as: artificial neural networks, fuzzy systems, decision trees, etc. Then, they can be used for optimization of parameters and structures of mentioned methods. It is worth to mention that, in the optimization process an extensive objective function (called also fitness function or evaluation function) can be used. Thanks to that, the quality criteria (e.g. related with accuracy and interpretability) as well as quantitative criteria (e.g. related with complexity) can be included in evaluation of the individuals. Such an approach was contemplated in previous research [17].

Currently, many varieties of population-based algorithms exist and they can be divided to: (a) single-population algorithms (see e.g. [22]) and multi-population algorithms (see e.g. [17]), (b) algorithms generating single solutions (see e.g. [1]) and fronts of solutions (see e.g. [8]), (c) algorithms using single-objective function (see e.g. [24]), and multi-objective function (see e.g. [5, 11]). The efficiency of those algorithms strongly depends on evolutionary operators used for exploration and exploitation of the search space (search space determines acceptable boundaries for parameters of solutions). Evolutionary operators usually have a set of parameters that must be selected before starting the evolution process (this is done usually by trial and error method) or modified in this process (usually by specified).

In this paper we have attempted to construct an algorithm that automatically selects evolutionary operators (for exploration and exploitation of search space) and their parameters. The idea behind this solution is that each individual in the population has (beside typical list of parameters) a list of available (typical) evolutionary operators and list of their parameters. At the beginning of the algorithm's operation, this list is initialized randomly, however while evolution of population it is updated simultaneously with typical parameters. In such approach individuals are modified basing on the values of list of available evolutionary operators and their parameters. In our previous work [16] this solution was partially considered. The problem was the large number of evolutionary operators used by the population. Due to that, the following improvements were made: (a) the list of available operators was improved, and (b) objective function was changed to reduce the number of used evolutionary operators. As a result, the individuals of the population that use fewer operators are better rated. Proposed approach has been used for nonlinear modeling using fuzzy systems.

The structure of this paper is as follows: in Sect. 2 a description of proposed method is placed, Sect. 3 contains simulation results and in Sect. 4 the conclusions are drawn.

2 Description of Proposed Method

Characteristics of the proposed method can be summarized as follows:

– The method is characterized by the ability of an automatic creation of model during the process of evolution. The most important feature of the method is the fact that during the process of evolution the selection and configuration of the evolution operators used for exploration and exploitation takes place simultaneously. This eliminates the need for selection of the type of operators and their parameters by trial and error method. This approach also allows to achieve an appropriate balance between exploration and exploitation of the search space.
– The method is based on the capabilities of fuzzy systems, which are a convenient tool for nonlinear modeling. Modeling can be done in different ways: (a) fuzzy system can model input-output dependences (see e.g. [9]), (b) fuzzy system can model elements of state variables matrix (see e.g. [3]), (c) fuzzy system can provide

appropriate cooperation of partial models representing various operating states of the modeled object (see e.g. [10]). In this paper the first type of modeling will be considered, and the description of system used in the modeling is given in Sect. 2.1.

– The method uses hybrid approach introduced by us earlier, enabling the simultaneous selection of real parameters and binary parameters (see e.g. [17]). Due to this, the structure and parameters used for modeling of fuzzy system and used in the process of evolution operators can be automatically selected.
– The method using fitness function component that promotes individuals that use lower number of evolutionary operators. Such solution not only affect the computational time but also causes the algorithm to work consistently.

The proposed method is based on the approach analogical to particle swarm optimization (PSO, [13]). PSO algorithm uses population of individuals, in which each individual encodes parameters of potential solution to the problem under consideration (marked as $\mathbf{X}_{ch}^{\mathrm{par}} = \{X_{ch,1}^{\mathrm{par}}, \ldots, X_{ch,L^{\mathrm{par}}}^{\mathrm{par}}\}$), velocity vector, used for modification of potential solution parameters ($\mathbf{X}_{ch}^{\mathrm{vel}} = \{X_{ch,1}^{\mathrm{vel}}, \ldots, X_{ch,L^{\mathrm{par}}}^{\mathrm{vel}}\}$), and best found so far parameters of potential solution **p** (marked as $\mathbf{X}_{ch}^{\mathrm{bst}} = \{X_{ch,1}^{\mathrm{bst}}, \ldots, X_{ch,L^{\mathrm{par}}}^{\mathrm{bst}}\}$). In the algorithm the best solution found in population is additionally remembered (marked as $\mathbf{X}^{\mathrm{glb}} = \{X_1^{\mathrm{glb}}, \ldots, X_{L^{\mathrm{par}}}^{\mathrm{glb}}\}$). In the evolution process, the parameters $\mathbf{X}_{ch}^{\mathrm{vel}}$ and $\mathbf{X}_{ch}^{\mathrm{par}}$ are subject to modification according to the following equation:

$$\begin{cases} X_{ch,g}^{\mathrm{vel}} := w \cdot X_{ch,g}^{\mathrm{vel}} + c_1 \cdot \mathrm{U}(0,1) \cdot \left(X_{ch,g}^{\mathrm{bst}} - X_{ch,g}^{\mathrm{par}}\right) + c_2 \cdot \mathrm{U}(0,1) \cdot \left(X_g^{\mathrm{glb}} - X_{ch,g}^{\mathrm{par}}\right) \\ X_{ch,g}^{\mathrm{par}} := X_{ch,g}^{\mathrm{par}} + X_{ch,g}^{\mathrm{vel}}, \end{cases} \tag{1}$$

where $ch = 1, \ldots, Npop$ is index of individual in population, $Npop$ stands for number of individuals in population, $g = 1, \ldots, L^{\mathrm{par}}$ is index of real parameter (gene), L^{par} stands for number of real parameters (actual number of parameters is different, which is explained in Sect. 2.2), w is inertia weight (usually $w \in [0.8, 1.0]$), c_1 and c_2 are cognitive and social parameters (see [13]), and $\mathrm{U}(a,b)$ is function returning random number from range $[a,b]$.

On the basis of PSO algorithm a generalization of (1) was proposed. The generalization was designed in a way that any evolutionary operator of exploration and exploitation can be used. Moreover, operators and their parameters can be selected dynamically in evolution process. This is consistent with two facts on the population algorithms: (a) in some algorithms operators of exploration and exploitation are intermingled, (b) in modification of parameters multiple operators can be used simultaneously. The generalized form of Eq. (1) takes the following form:

$$\begin{cases} X_{ch,g}^{\mathrm{vel}} := w \cdot X_{ch,g}^{\mathrm{vel}} + \sum\limits_{o=1}^{L^{\mathrm{op}}} X_{ch,o}^{\mathrm{op}} \cdot \mathrm{op}_o\left(X_{ch,g}^{\mathrm{par}}, X_g^{\mathrm{glb}}, X_{ch,g}^{\mathrm{bst}}, X_{ch1,g}^{\mathrm{par}}, X_{ch2,g}^{\mathrm{par}}, X_{ch,g}^{\mathrm{imp}}\right) \\ X_{ch,g}^{\mathrm{par}} := X_{ch,g}^{\mathrm{par}} + X_{ch,g}^{\mathrm{vel}}, \end{cases}, \tag{2}$$

where vector $\mathbf{X}_{ch}^{\mathrm{op}} = \{X_{ch,1}^{\mathrm{op}}, \ldots, X_{ch,L^{\mathrm{op}}}^{\mathrm{op}}\}$ contains information on binary keys that stand for activation state of operators connected with them (an assumption was taken that 0 value stands for excluded operator and vice versa), L^{op} stands for number of considered operators (see Table 1), $\mathrm{op}_o(\cdot)$ stand for functions representing operator with index o, $\mathbf{X}_{ch1}^{\mathrm{par}}$ and $\mathbf{X}_{ch2}^{\mathrm{par}}$ stand for parent individuals chosen by selection method (for example roulette wheel method, [20]), $\mathbf{X}_{ch}^{\mathrm{imp}}$ means additional individual used in assimilation operator (Table 1, $o = 10$, [2]). The individual $\mathbf{X}_{ch}^{\mathrm{imp}}$ stands for imperialist of current solution selected as follows:

$$X_{ch}^{\mathrm{imp}} = \begin{cases} X_{ch,g}^{\mathrm{par}} & \text{for } ch \leq Nimp \\ X_{Nimp\cdot((ch-Nimp)/(Npop-Nimp)),g}^{\mathrm{par}} & \text{for } ch > Nimp \end{cases}, \tag{3}$$

where $Nimp$ stands for number of empires [2].

The set of considered evolutionary operators is presented in Table 1. In this table the following marks are additionally used: $\alpha = \mathrm{U}(0,1)$ stands for a random number generated individually for each parameter under modification, $\beta = \mathrm{U}(0,1)$ stands for a random number generated individually for each individual under modification, $RInd$ is randomly chosen index of parameter, $RSet$ contains a set of randomly chosen indexes of parameters, ff(.) stand for objective function of individual, ff_{min} is a smallest value of objective function for current population, $\mathrm{U}_G(1,1)$ is random number according to Gaussian distribution with mean 1 and variance 1, A^t is parameter additionally multiplied by coefficient α^t each time when individual modification improves solution (see e.g. [24]), $dist(\mathbf{X}_{ch1}, \mathbf{X}_{ch})$ stands for Manhattan distance between genes of two individuals.

2.1 Fuzzy System Used for Nonlinear Modeling

As previously mentioned, a multi-input, multi-output fuzzy system of the Mamdani-type that maps $\mathbf{X} \to \mathbf{Y}$ is used, where $\mathbf{X} \subset \mathbf{R}^n$ and $\mathbf{Y} \subset \mathbf{R}^m$ is used for nonlinear modeling. The fuzzy rule base of the system consists of a collection of N fuzzy if-then rules that takes the following form:

$$R^k : \left[\mathrm{IF}\left(x_1 \text{ is } A_1^k\right)\mathrm{AND}\ldots\mathrm{AND}\left(x_n \text{ is } A_n^k\right)\mathrm{THEN}\left(y_1 \text{ is } B_1^k\right),\ldots,\left(y_m \text{ is } B_m^k\right)\right], \tag{4}$$

where $\mathbf{x} = [x_1, \ldots, x_n] \in \mathbf{X}$, $\mathbf{y} = [y_1, \ldots, y_m] \in \mathbf{Y}$, $A_1^k, \ldots, A_n^k$ are fuzzy sets characterized by membership functions $\mu_{A_i^k}(x_i)$, $i = 1, \ldots, n$, $k = 1, \ldots, N$, n stands for number of inputs, $B_1^k, \ldots, B_m^k$ are fuzzy sets characterized by membership functions $\mu_{B_j^k}(y_j)$, $j = 1, \ldots, m$, $k = 1, \ldots, N$, and m stands for number of outputs. In this paper a Gaussian membership type functions are considered [20].

In the Mamdani approach (see e.g. [20]) output signal $\bar{y}_j$, $j = 1, \ldots, m$, of the fuzzy system is described by the following formula (for more details see [9]):

$$\bar{y}_j = \frac{\sum_{r=1}^{R} \bar{y}_{j,r}^{\mathrm{def}} \cdot \underset{k=1}{\overset{N}{S}} \left\{ T\left\{ \underset{i=1}{\overset{n}{T}} \left\{ \mu_{A_i^k}(\bar{x}_i) \right\}, \mu_{B_j^k}\left(\bar{y}_{j,r}^{\mathrm{def}}\right) \right\} \right\}}{\sum_{r=1}^{R} \underset{k=1}{\overset{N}{S}} \left\{ T\left\{ \underset{i=1}{\overset{n}{T}} \left\{ \mu_{A_i^k}(\bar{x}_i) \right\}, \mu_{B_j^k}\left(\bar{y}_{j,r}^{\mathrm{def}}\right) \right\} \right\}}, \tag{5}$$

where $\bar{y}^{\text{def}}_{j,r}$, $j = 1,\ldots,m$, $r = 1,\ldots,R$, are discretization points, R is a number of discretization points, $T\{\cdot\}$ is a t-norm, and $S\{\cdot\}$ is a t-conorm (see e.g. [15]).

2.2 Encoding of the Individuals

The selected encoding refers to Pittsburgh approach [12]. Thus, the single individual $\mathbf{X}_{ch}$ encodes the information on:

- Keys of operators (binary parameters). They are encoded as $\mathbf{X}^{\text{op}}_{ch}$ and they are used in Eq. (2) (see Table 1):

$$\mathbf{X}^{\text{op}}_{ch} = \left\{op_{ch,1},\ldots,op_{ch,L^{op}}\right\} = \left\{X^{\text{op}}_{ch,1},\ldots,X^{\text{op}}_{ch,L^{op}}\right\}, \tag{6}$$

  where L^{op} stands for number of evolutionary operators that can be used by individual encoded as $\mathbf{X}_{ch}$.
- Parameters of fuzzy system (5) and parameters of operators (for the list of operators parameters see Table 1):

$$\mathbf{X}^{\text{par}}_{ch} = \left\{\begin{array}{c} \bar{x}^A_{ch,1,1}, \sigma^A_{ch,1,1},\ldots,\bar{x}^A_{ch,n,1}, \sigma^A_{ch,n,1},\ldots \bar{x}^A_{ch,1,N}, \sigma^A_{ch,1,N},\ldots,\bar{x}^A_{ch,n,N}, \sigma^A_{ch,n,N}, \\ \bar{y}^B_{ch,1,1}, \sigma^B_{ch,1,1},\ldots,\bar{y}^B_{ch,m,1}, \sigma^B_{ch,m,1},\ldots \bar{y}^B_{ch,1,N}, \sigma^B_{ch,1,N},\ldots,\bar{y}^B_{ch,m,N}, \sigma^B_{ch,m,N}, \\ \bar{y}^{\text{def}}_{ch,1,1},\ldots,\bar{y}^{\text{def}}_{ch,1,R},\ldots,\bar{y}^{\text{def}}_{ch,m,1},\ldots,\bar{y}^{\text{def}}_{ch,m,R}, \\ w, c_1, c_2, p_c, m_r, p_m, F^{\text{DE}}, d, \alpha^{\text{FA}}, \beta^{\text{FA}}, \gamma, CR, f_{\min}, f_{\max}, A^t, \alpha^t, \hat{A}, F^{\text{BTO}} \end{array}\right.$$
$$= \left\{X^{\text{par}}_{ch,1},\ldots,X^{\text{par}}_{ch,L^{\text{par}}}\right\}, \tag{7}$$

  where $\{\bar{x}^A_{i,k}, \sigma^A_{i,k}\}$ stand for parameters of membership functions of input Gaussian fuzzy sets $A^k_1,\ldots,A^k_n$, $\{\bar{y}^B_{j,k}, \sigma^B_{j,k}\}$ stand for parameters of membership functions of output Gaussian fuzzy sets $B^k_1,\ldots,B^k_m$, and L^{par} is the number of genes of part $\mathbf{X}^{\text{par}}_{ch}$.
- Velocity vector of parameters $\mathbf{X}^{\text{par}}_{ch}$ (see (7)). They are encoded as set $\mathbf{X}^{\text{vel}}_{ch}$.
- Best location of individual $\mathbf{X}^{\text{bst}}_{ch}$ (they are copy of best parameters of $\mathbf{X}^{\text{par}}_{ch}$).

A individual is thus a collection of components: $\mathbf{X}_{ch} = \{\mathbf{X}^{\text{op}}_{ch}, \mathbf{X}^{\text{par}}_{ch}, \mathbf{X}^{\text{vel}}_{ch}, \mathbf{X}^{\text{bst}}_{ch}\} = \{X_{ch,1},\ldots,X_{ch,L}\}$ with a total length of genes equal to $L = 3 \cdot L^{\text{par}} + L^{\text{op}}$.

2.3 Evaluation of the Individuals

The aim of the objective function (fitness function) is to minimize following error:

$$\text{ff}(\mathbf{X}_{ch}) = T\{\epsilon(\mathbf{X}_{ch}), \delta(\mathbf{X}_{ch})\}. \tag{8}$$

Table 1. Chosen functions that represent evolutionary operators of exploration and exploitation considered in this paper.

o	Base method	$\text{op}_o\left(X^{\text{par}}_{ch,g}, X^{\text{glb}}_{g}, X^{\text{bst}}_{ch,g}, X^{\text{par}}_{ch1,g}, X^{\text{par}}_{ch2,g}, X^{\text{imp}}_{ch,g}\right)$	Parameters
1	PSO-best [13]	$c_1 \cdot \text{U}(0,1) \cdot \left(X^{\text{bst}}_{ch,g} - X^{\text{par}}_{ch,g}\right)$	$c_1 \in [1.5, 2.5]$
2	PSO-global [13]	$c_2 \cdot \text{U}(0,1) \cdot \left(X^{\text{glb}}_{g} - X^{\text{par}}_{ch,g}\right)$	$c_2 \in [1.5, 2.5]$
3	GA-crossover [20]	$\begin{cases} \text{U}(0,1) \cdot \left(X^{\text{par}}_{ch1,g} - X^{\text{par}}_{ch,g}\right) & \text{for} \quad \beta < p_c \\ 0 & \text{for} \quad \text{otherwise} \end{cases}$	$p_c \in [0.7, 1.0]$
4	GA-mutation [20]	$\begin{cases} \text{U}(-1,1) \cdot m_r & \text{for} \quad \alpha < p_m \\ 0 & \text{for} \quad \text{otherwise} \end{cases}$	$m_r \in [0.01, 0.20]$, $p_m \in [0.05, 0.50]$
5	DE-crossover [1]	$\begin{cases} F^{\text{DE}} \cdot \left(X^{\text{par}}_{ch1,g} - X^{\text{par}}_{ch2,g}\right) \\ \text{for} \quad (\alpha < CR) \text{ or } (ch = RInd) \\ 0 \quad \text{for} \quad \text{otherwise} \end{cases}$	$F^{\text{DE}} \in [0, 2]$, $CR \in [0, 1]$
6	BAT-movement [24]	$\text{U}(f_{\min}, f_{\min} + f_{\max}) \cdot \left(X^{\text{glb}}_{g} - X^{\text{par}}_{ch,g}\right)$	$f_{min} \in [0.0, 0.5]$, $f_{max} \in [0.0, 1.0]$
7	BAT-walk [24]	$\text{U}(-1,1) \cdot \sum_{ch3=1}^{Npop} \frac{A^t_{ch3}}{Npop}$	$A^t \in [0.0, 0.5]$, $\alpha^t \in [0.9, 1.0]$
8	FWA-explosion [22]	$\begin{cases} \frac{\text{U}(-1,1) \cdot \hat{A} \cdot (\text{ff}(\mathbf{X}_{ch}) - \text{ff}_{\min})}{\sum_{ch3=1}^{Npop} (\text{ff}(\mathbf{X}_{ch3}) - \text{ff}_{\min})} & \text{for} \quad ch \in RSet \\ 0 & \text{for} \quad \text{otherwise} \end{cases}$	$\hat{A} \in [0.1, 2.0]$
9	FWA-mutation [22]	$\begin{cases} X^{\text{par}}_{ch,g} \cdot \text{U}_G(1,1) - X^{\text{par}}_{ch,g} & \text{for} \quad ch \in RSet \\ 0 & \text{for} \quad \text{otherwise} \end{cases}$	
10	ICA-assimilation [2]	$d \cdot U(0,1) \cdot \left(X^{\text{imp}}_{ch,g} - X^{\text{par}}_{ch,g}\right)$	$d \in [0.5, 2]$
11	FA-movement [23]	$\begin{cases} \beta^{\text{FA}} \cdot \left(X^{\text{par}}_{ch1,g} - X^{\text{par}}_{ch,g}\right) \cdot \\ \cdot e^{-\gamma \cdot dist(\mathbf{X}_{ch1}, \mathbf{X}_{ch})^2} & \text{for} \quad \left(\begin{matrix} \text{ff}(\mathbf{X}_{ch1}) < \\ \text{ff}(\mathbf{X}_{ch}) \end{matrix}\right) \\ 0 & \text{for} \quad \text{otherwise} \end{cases}$	$\beta^{\text{FA}} \in [0.9, 1.1]$, $\gamma \in [0.01, 100]$
12	FA-walk [23]	$\alpha^{\text{FA}} \cdot U\left(-\frac{1}{2}, \frac{1}{2}\right)$	$\alpha^{\text{FA}} \in [0.01, 0.2]$

The component $\epsilon(\mathbf{X}_{ch})$ stands for standardized error of nonlinear modeling calculated as follows:

$$\epsilon(\mathbf{X}_{ch}) = \frac{1}{m} \cdot \sum_{j=1}^{m} \left(\frac{1}{Z} \cdot \sum_{z=1}^{Z} \left| d_{z,j} - \bar{y}_{z,j} \right| \right) \Big/ \left(\max_{z=1,\ldots,Z} \{d_{z,j}\} - \min_{z=1,\ldots,Z} \{d_{z,j}\} \right), \tag{9}$$

where $d_{z,j}$ is expected value of j-th output for z-th data sample ($z = 1, \ldots, Z$), Z stands for number of data samples, $\bar{y}_j$ is fuzzy system (5) output value calculated for data sample $\bar{\mathbf{x}}_z$. The purpose of normalization is to eliminate the differences between the errors of different outputs of the system (5) in case where $m > 1$.

The component $\delta(\mathbf{X}_{ch})$ stands for penalty of using too few or too many operators:

$$\delta(\mathbf{X}_{ch}) = \frac{1}{L^{op} - Nop}\left|\sum_{o=1}^{L^{op}} op_{ch,o} - Nop\right|, \tag{10}$$

where $Nop \geq 1$ stands for desired number of active operators.

2.4 Processing of the Individuals

The proposed method for processing the population taking into account the Eq. (2) allows use of any combination of evolutionary operators. This process is executed according to the following steps:

– Step 1. Initialization of population. In this step all genes of all *Npop* individuals are set randomly. It takes into account the specification of the problem under consideration and operators parameters ranges stated in Table 1. In case of genes $\mathbf{X}_{ch}^{\text{vel}}$ the assumptions that the genes do not exceed 20% ranges of the corresponding genes from $\mathbf{X}_{ch}^{\text{par}}$ are additionally taken.
– Step 2. Evaluation of population. This step aims to evaluate each individual by using fitness function (9).
– Step 3. Reproduction. It is based on mechanisms analogical to algorithm PSO (with improvements that makes generalization possible):
  - For each individual $\mathbf{X}_{ch}$ a copy is created (which allows to use of any selection method-see Step 4).
  - Binary genes of the copy $\mathbf{X}_{ch}^{\text{op}}$ are modified using crossover and mutation operators from genetic algorithm [20]. Parameters of these operators (p_m and p_c) are encoded in set $\mathbf{X}_{ch}^{\text{par}}$.
  - Real genes of the copy $\mathbf{X}_{ch}^{\text{vel}}$ and $\mathbf{X}_{ch}^{\text{par}}$ are updated according to (2). Genes $\mathbf{X}_{ch}^{\text{par}}$ are also repaired (narrowed down to specified boundaries).
  - The copies of $\mathbf{X}_{ch}$ are evaluated by fitness function (9). If the fitness function value of the copy is better than the fitness function value of the individual, the genes $\mathbf{X}_{ch}^{\text{bst}}$ of the copy are set as genes $\mathbf{X}_{ch}^{\text{par}}$ of the copy.
– Step 4. New generation selection. Proposed approach allows to use different strategy for selection new population individuals:
  - Strategy S1. In this strategy, the copy of an individual replaces the individual (as in the algorithm PSO).
  - Strategy S2. In this strategy, the copy of an individual replaces the individual only if the fitness function value of the copy is better than the fitness function value of the individual (similar to BAT algorithm [24]).
  - Strategy S3. In this strategy individuals and their modified copies are placed in temporary population, and then the best (according to fitness function) *Npop* individuals are selected for next generation.
– Step 5. Stop condition. In this step a stop condition is met (for example if the specified number of algorithm iterations is achieved). If this condition is not met, the algorithm goes back to Step 3, otherwise the best solution is presented and algorithm stops.

3 Simulations

The goals of the simulations were to test:

- Classic algorithm PSO ($X^{\mathrm{op}}_{ch,g} = 1$, $g = 0, 1$, and $X^{\mathrm{op}}_{ch,g} = 0$, $g = 2, 3, \ldots 11$), algorithm PSO enriched by using other evolutionary operators (PSO-OP) ($X^{\mathrm{op}}_{ch,g} = 1$, $g = 0, 1$, and $X^{\mathrm{op}}_{ch,g} \in \{0, 1\}$, $g = 2, 3, \ldots 11$) and proposed algorithm with flexible selectable evolutionary operators (OP) ($X^{\mathrm{op}}_{ch,g} \in \{0, 1\}$, $g = 0, 1, \ldots 11$).
- Different strategies S1-S2 for selecting next generation of individuals, described in Sect. 2.4.

Table 2. Nonlinear benchmarks used in simulations.

Item no.	Problem name and reference	Short name	Number of inputs (n)	Number of outputs (m)	Number of data rows (Z)
1.	Auto MPG [19]	ampg	7	1	398
2.	Chemical Plant [21]	cplant	3	1	70
3.	Computer Hardware [14]	cpu	9	1	209
4.	Concrete Slump Test [7]	slump	7	3	103
5.	Yacht Hydrodynamics [18]	yacht	6	1	308

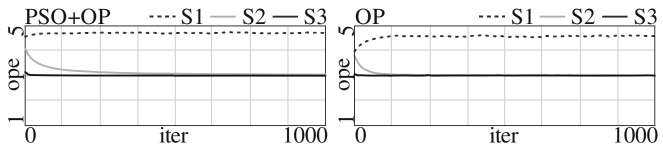


Fig. 1. Average number of used operators by single individuals (ope) averaged for all considered benchmarks. Dotted line stands for strategy S1, light line stands for strategy S2, and dark line stands for strategy S3.

The simulations were performed for typical benchmarks from nonlinear modeling field (see Table 2). The parameters of simulations were set as follows: number of fuzzy rules $N = 3$, number of discretization points $R = 3$, triangular norms: algebraic, number of individuals $Npop = 100$, number of empires $Nimp = 10$, number of expected operators $Nop = 3$, number of iterations: 1000, number of repeats of simulations for each problem and case: 20, selection method for choosing parents of the individual: roulette wheel.

The average simulation results are presented in Tables 3 and 4. The average number of used operators is shown in Figs. 2 and 3. The average number of used operators per individual is shown in Fig. 1.

The simulations conclusions are as follows:

- The best results in terms of the fitness function was obtained for OP method, regardless of the used strategy (see Table 3). This has a bearing on the results in terms of RMSE (see Table 4).

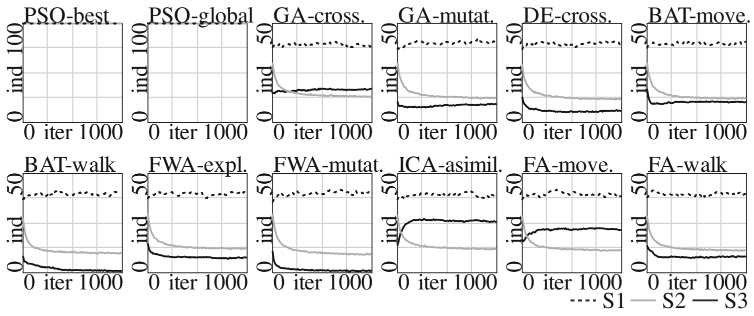


Fig. 2. Average number of individuals (ind) that use specified operators for algorithm PSO+OP averaged for all considered benchmarks. Dotted line stands for strategy S1, light line stands for strategy S2, and dark line stands for strategy S3.

Table 3. Averaged values of ff(.) for considered benchmarks. Best results for each strategy are shown in bold.

Strategy	Method	ampg	cplant	cpu	slump	yacht	Avg.
S1	PSO	0.0747	0.0891	0.0397	0.0788	0.0876	0.0740
	PSO+OP	0.0751	0.0798	0.0295	0.0782	0.0765	0.0678
	OP	**0.0546**	**0.0483**	**0.0190**	**0.0672**	**0.0531**	**0.0484**
S2	PSO	0.0822	0.0792	0.0458	0.0780	0.0900	0.0750
	PSO+OP	0.0497	0.0350	0.0174	0.0586	0.0398	0.0401
	OP	**0.0336**	**0.0081**	**0.0090**	**0.0435**	**0.0247**	**0.0238**
S3	PSO	0.0747	0.0679	0.0307	0.0762	0.0827	0.0664
	PSO+OP	0.0468	0.0289	0.0172	0.0551	0.0424	0.0381
	OP	**0.0330**	**0.0052**	**0.0066**	**0.0312**	**0.0255**	**0.0203**

Table 4. Averaged values of $\epsilon(\mathbf{X}_{ch})$ for considered benchmarks. Best results for each strategy are shown in bold.

Strategy	Method	ampg	cplant	cpu	slump	yacht	Avg.
S1	PSO	0.1383	0.1671	0.0515	0.1465	0.1641	0.1335
	PSO+OP	0.1224	0.1410	0.0313	0.1378	0.1437	0.1152
	OP	**0.0813**	**0.0689**	**0.0194**	**0.1066**	**0.1062**	**0.0765**
S2	PSO	0.1643	0.0843	0.0730	0.1561	0.1523	0.1260
	PSO+OP	0.0993	0.0701	0.0348	0.1173	0.0795	0.0802
	OP	**0.0671**	**0.0161**	**0.0179**	**0.0871**	**0.0494**	**0.0475**
S3	PSO	0.1494	0.0895	0.0613	0.1525	0.1470	0.1199
	PSO+OP	0.0935	0.0578	0.0344	0.1101	0.0848	0.0761
	OP	**0.0661**	**0.0104**	**0.0132**	**0.0624**	**0.0509**	**0.0406**

- The best results in terms of used fitness function were obtained for strategy S3, regardless of the used method. Strategy S1 used the largest number of operators, strategy S2 gradually reduced the number of used operators, while strategy S3 smoothly adjusted the number of used operators during the evolution of population (see Figs. 2 and 3).
- The best results was obtained for OP method and strategy S3 (see Tables 3 and 4). Such an approach allowed a dynamic matching of the number and type of used operators during the evolution of a population (see Fig. 3).
- The least used operators for OP method and strategy S3 were: PSO-best, BAT-walk, and FWA-mutation (after 50 iterations of algorithm, the usage of these operators has dropped to almost zero - see Fig. 3).
- The most used operators for OP method and strategy S3 were: DE-crossover, BAT-move, and FA-move (number of used operators increased during the evolution of population - see Fig. 3).
- The use of S1 strategy led to use over 4 operators by each individual in the population (see Fig. 1). For other strategies, the number of used operators sought to the *Nop* value. For the S2 strategy it was achieved after about 150 iterations (see Fig. 1), and for strategy S3 after about 30 iterations.

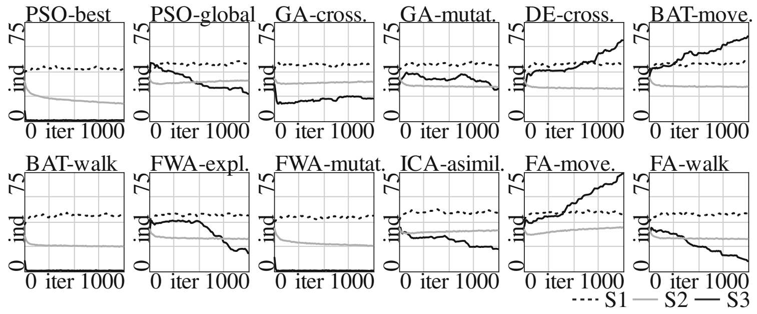


Fig. 3. Average number of individuals (ind) that use specified operators for algorithm OP averaged for all considered benchmarks. Dotted line stands for strategy S1, light line stands for strategy S2, and dark line stands for strategy S3.

4 Conclusions

The proposed in this paper population-based algorithm for nonlinear modeling with the selection of evolutionary operators allows, among others, for: (a) a precise control of the mechanisms of exploration and exploitation, (b) elimination of the need for selection of evolutionary operators by trial and error method, (c) reduction of the number of evolutionary operators, and (d) speeding up the optimization. This allows the algorithm to work well with other methods, especially fuzzy systems, for solving nonlinear modeling problems. Simulation studies have confirmed the effectiveness of presented approach.

Acknowledgment. The project was financed by the National Science Centre (Poland) on the basis of the decision number DEC-2012/05/B/ST7/02138.

The authors would like to thank the reviewers for very helpful suggestions and comments in the revision process.

References

1. Ali, M., Pant, M., Abraham, A.: Unconventional initialization methods for differential evolution. Appl. Math. Comput. **219**(9), 4474–4494 (2013)
2. Atashpaz-Gargari, E., Lucas, C.: Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. IEEE Congr. Evol. Comput. **7**, 4661–4666 (2007)
3. Bartczuk, Ł., Przybył, A., Koprinkova-Hristovak P.: New method for nonlinear fuzzy correction modelling of dynamic objects. Lecture Notes in Computer Science, vol. 8467, pp. 169–180 (2014)
4. Borzemski, L., Kliber, M., Nowak, Z.: Using data mining algorithms in web performance prediction. Cybern. Syst. Int. J. **40**(2), 176–187 (2009)
5. Brester, C., Semenkin, E., Sidorov, M.: Multi-objective heuristic feature selection for speech-based multilingual emotion recognition. J. Artif. Intell. Soft Comput. Res. **6**(4), 243–253 (2016)
6. Che, J., Wang, J., Li, K.: A Monte Carlo based robustness optimization method in new product design process: a case study. Am. J. Ind. Bus. Manag. **4**(7), 360–369 (2014)
7. Cheng, Y.I.: Modeling slump flow of concrete using second-order regressions and artificial neural networks. Cem. Concr. Compos. **29**(6), 474–480 (2007)
8. Corne, D.W., Knowles, J.D., Oates, M.J.: The Pareto envelope-based selection algorithm for multiobjective optimisation. In: Schoenauer, S., et al. (eds.) Parallel Problem Solving from Nature – PPSN VI, pp. 839–848. Springer, Berlin (2000)
9. Cpałka, K., Łapa, K., Przybył, A., Zalasiński, M.: A new method for designing neuro-fuzzy systems for nonlinear modelling with interpretability aspects. Neurocomputing **135**, 203–217 (2013)
10. Dziwiński, P., Bartczuk, Ł., Przybył, A., Avedyan, E.D.: A new algorithm for identification of significant operating points using swarm intelligence. Artificial Intelligence and Soft Computing, vol. 8468, pp. 349–362 (2014)
11. Ehrgott, M.: Multicriteria Optimization. Springer Verlag, Heidelberg (2000)
12. Ishibuchi, H., Nakashima, T., Murata, T.: Comparison of the Michigan and Pittsburgh approaches to the design of fuzzy classification systems. Electron. Commun. Jpn. **80**, 10–19 (1997)
13. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of the IEEE International Conference on Neural Networks, pp. 1942–1948 (1995)
14. Kibler, D., Aha, D.: Instance-based prediction of real-valued attributes. In: Proceedings of the CSCSI (Canadian AI) Conference, vol. 5, pp. 51–57 (1988)
15. Klement, E.P., Mesiar, R., Pap, E.: Triangular Norms. Springer, Heidelberg (2000)
16. Łapa, K., Cpałka, K., Wang, L.: A method for nonlinear fuzzy modelling using population based algorithm with flexibly selectable operators. Artificial Intelligence and Soft Computing, Lecture Notes in Computer Science. Springer (2017) (in print)
17. Łapa, K., Szczypta, J., Venkatesan, R.: Aspects of structure and parameters selection of control systems using selected multi-population algorithms. Artificial Intelligence and Soft Computing, vol. 9120, pp. 247–260 (2015)

18. Ortigosa, I., Lopez, R., Garcia, J.: A neural networks approach to residuary resistance of sailing yachts prediction. In: Proceedings of the International Conference on Marine Engineering (2007)
19. Quinlan, R.: Combining instance-based and model-based learning. In: Proceedings of the Tenth International Conference of Machine Learning, pp. 236–243 (1993)
20. Rutkowski, L.: Computational Intelligence. Methods and Techniques. Springer-Verlag, Heidelberg (2008)
21. Sugeno, M., Yasukawa, T.: A fuzzy-logic-based approach to qualitative modeling. IEEE Trans. Fuzzy Syst. **1**(1), 7–31 (1993)
22. Tan, Y., Zhu, Y.: Fireworks algorithm for optimization. In: Advances in Swarm Intelligence, pp. 355–364 (2010
23. Yang, X.S.: Nature-Inspired Metaheuristic Algorithms. Luniver Press, Bristol (2008)
24. Yang, X.S.: A new metaheuristic bat-inspired algorithm. In: Nature Inspired Cooperative Strategies for Optimization, NICSO 2010. SCI, vol. 284, pp. 65–74. Springer (2010)

Evaluation of Gated Recurrent Neural Networks in Music Classification Tasks

Jan Jakubik(✉)

Department of Computational Intelligence,
Wrocław University of Science and Technology, Wrocław, Poland
jan.jakubik@pwr.edu.pl

Abstract. In this paper, we evaluate two popular Recurrent Neural Network (RNN) architectures employing the mechanism of gating: Long-Short Term Memory (LSTM) and Gated Recurrent Unit (GRU), in music classification tasks. We examine the performance on four datasets concerning genre, emotion and dance style recognition. Our key result is a significant improvement of classification accuracy achieved by training the recurrent network on random short subsequences of the vector sequences in the training set. We examine the effect of this training approach on both architectures and discuss the implications for the potential use of RNN in music information retrieval.

Keywords: Artificial intelligence · Machine learning · Recurrent Neural Network · Music Information Retrieval

1 Introduction

Music Information Retrieval (MIR) is a research area of growing relevance due to the popularization of online music services and an increasing number of unannotated audio files available online. Searching and automatically annotating music files is a complex task, which requires approaching music simultaneously from multiple angles: one must consider genres, emotions, specific artists, instruments, etc. A number of separate machine learning problems have been formulated due to this: genre classification, emotion classification, emotion regression, generalized annotation, etc.

Recent advancements in neural networks are allowing us to explore sequential data more efficiently than before, so far showing good results in domains of text [1] and speech analysis [2]. Specifically, Long Short Term Memory networks [3] which are designed to eliminate vanishing and exploding gradients allowed significant advancements. A recurrent neural network approach can benefit music information retrieval in two ways: firstly, it enables the possibility of deeper architectures, which are known to work well with low-level representations of data, therefore eliminating the problem of designing new domain-specific features. Secondly, it allows modeling time dependencies in sequential data, which are of key importance in certain aspects of music such as tempo and rhythm.

However, it is debated whether Long Short Memory networks are actually the best possible network architecture for the tasks they are designed to perform. Multiple simplified architectures [4] where introduced and shown to achieve comparable performance. Comparative studies of these architectures exist, but they focus neither on music audio nor on classification of entire sequences.

In this paper, we evaluate two most well-known recurrent network architectures with gating mechanisms on four music classification datasets. The goal of our research is to establish which of the available recurrent neural network models can be used efficiently in music classification problems, given the typical properties of MIR datasets. As we expect the length of vector sequences to cause a problem with training, we attempt to train on shorter excerpts of the training set music files. We demonstrate the influence of this approach on the performance of both LSTM and GRU, and discuss the implications of our results.

2 Related Work

Our work is related to the Music Information Retrieval research and other comparative studies of recurrent neural networks. Below we summarize the related literature for both of the subjects.

2.1 Music Audio Classification

Music audio classification is a problem relevant to music retrieval and auto-tagging. Single-label classification datasets are typically crafted for a specific task, such as genre recognition or emotion recognition, allowing researchers to focus on a single type of potential tags. The importance of such focused research comes from the fact that the selection of features describing music can differ significantly for different types of tags. Features which typically describe emotions may be insufficient to describe genres and vice versa.

The task of classifying music genres was explored in-depth by many researchers. A variety of approaches has been employed, including stochastic times series models, sparse coding, and bag-of-features representations [5]. Researchers reported results as high as 90% accuracy [6] on a 10-label GTZAN dataset [7], the most popular set for music genre classification. However, these results may be considered controversial as the dataset itself has come under heavy criticism due to flaws in its construction. It was found [8] that GTZAN contains repeated audio excerpts, mislabeling and certain genres are biased towards a single artist. While new datasets have been created since, it is now hard to refer to existing literature without considering a possible influence of the aforementioned faults on the results.

The task of recognizing music emotions can be treated as either classification or regression, the latter being more popular due to the significance of Valence-Arousal model of emotions [9]. However, attempts to classify musical files with regards to emotion have been made [10]. In [11], the selection of features for mood recognition was explored. Music emotion recognition was defined as a classification task, and the class labels correspond to four quadrants of the Valence-Arousal plane, namely “angry”, “happy”, “relaxed” and “sad”. The labels for classification were obtained based on tags from the LastFM website. The accuracy of classification for this task is rather low, at 54% for the best performing feature set.

Other, less thoroughly explored classification problems include: instrument recognition [12], playing style recognition [13]. In this paper, we focus on the problems which require processing a long (30 s and more) musical excerpt. We are interested in new problems that may arise in such tasks, which may not have been present in the evaluation of RNN on data such as natural language datasets.

2.2 Gated Recurrent Neural Networks

The details of recurrent neural network architectures we employ in this paper are explained in Sect. 3. In this section, we summarize other papers which analyze these architectures and compare them to each other in varying tasks.

In [14] an empirical comparison between LSTM and GRU was made. The comparison involved tasks of symbolic music modeling and speech audio modeling. While results did not prove one architectures’ superiority over the other in terms of prediction error, a practical consideration that GRU involves fewer parameters than LSTM implies GRU is the better choice.

In [15] a large scale comparison including tasks of speech recognition, handwriting recognition, and polyphonic music modeling was performed on LSTM and variants thereof. The authors concluded that changes to the original LSTM architecture do not improve the performance significantly.

In [16], the authors employed a probabilistic search approach to explore the space of possible gated network architectures in an evolutionary-like fashion. Using a mutation operator which changes the structure of an RNN, the goal of the search was to find architectures performing better than GRU and LSTM. The architectures were evaluated on tasks of language modeling, XML modeling, and polyphonic music modeling. New mutated architectures were found to be able to outperform both GRU and LSTM, however, the LSTM and GRU performance was still good.

We find comparative studies mentioned above insufficient to draw conclusions regarding the usage of RNN in music classification tasks. There are two important properties of audio musical files which must be taken into account:

- dense inputs: a representation of music audio typically uses dense vectors. In contrast, language modeling tasks which are among the most popular problems to solve with LSTM may have extremely sparse inputs, e.g., a word occurrence vector in a dictionary with many thousands of words. Similarly, symbolic music modeling can use sparse inputs.
- length of the sequences: using a low-level representation such as spectrogram or cepstral coefficients, it is fairly standard to use a frame size of about 50 ms, with partial frame overlap. For a 30 s music file, which is a standard length of music excerpts in MIR datasets, this results in a sequence of 1200 vectors. This is longer than sequences which appear in other domains.

While symbolic music modeling appears among tasks considered in papers mentioned above, it is significantly different than the classification of actual audio recordings, due to different representations and the fact that a timeframe in which the music will be modeled can be freely chosen as a part of defining the task. I.e. one might consider the task of predicting the next note given 10, 20 or 30 previous notes, and use this as a basis of their performance comparison. In contrast, the type of music classification tasks we consider in this paper has its ground truth labels given for the entire music file. This forces us to aggregate the output of a recurrent layer over an entire music piece which may range from 30 s to a few minutes. As we will demonstrate in Sect. 4, the length of sequences we use as training samples has a significant impact on the classification results.

3 Gated Recurrent Network Architectures

Standard model of a recurrent neural network layer without gating mechanism [17] is given below in Eq. (1). For a series of l input vectors $X = (x_1, x_2, \ldots, x_l)$, the output vector h_t of the layer at time t is given by:

$$h_t = \sigma(Wx_t + Uh_{t-1} + b) \tag{1}$$

where h_{t-1} is the previous activation and x_t is the current input. σ is an activation function, applied to every element of a vector. We will use the following notation for activation functions: σ_{sig} denotes log-sigmoid, σ_{tanh} denotes hyperbolic tangent. Parameters of the model are weight matrices W, U and the bias vector b. The network can be unfolded in a non-recurrent one and trained with backpropagation.

Modern recurrent neural networks have improved the performance on long time series by extending the model above with the concept of gates. A gated network unit (which replaces a standard recurrent layer) can have many interconnected internal layers, and outputs of these layers can be multiplied element-wise. In practice, this makes the output of log-sigmoid layers function as "gates" which can pass the output of another layer (if the log-sigmoid activation is 1) or block it (if the log-sigmoid activation is 0). This principle was proposed in Long-Short Term Memory (LSTM) unit Eqs. (2–6):

$$f_t = \sigma_{sig}\left(W_f x_t + U_f h_{t-1} + b_f\right) \tag{2}$$

$$i_t = \sigma_{sig}\left(W_i x_t + U_i h_{t-1} + b_i\right) \tag{3}$$

$$o_t = \sigma_{sig}\left(W_o x_t + U_o h_{t-1} + b_o\right) \tag{4}$$

$$c_t = f_t * c_{t-1} + i_t * \sigma_{tanh}\left(W_c x_t + U_c h_{t-1} + b_c\right) \tag{5}$$

$$h_t = o_t * \sigma_{tanh}\left(c_t\right) \tag{6}$$

The outputs r_t, i_t and o_t represent gates, i.e. standard log-sigmoid recurrent layers, each with two corresponding weight matrices (W_r, U_r, W_i, U_i, W_o, U_o) and a bias vector

(b_r, b_i, b_o). The LSTM unit has an internal memory state c_t which requires another two weight matrices W_c, U_c and a bias vector b_c.

GRU [18] is a simplified variant of a gated model, defined by Eqs. (7–9):

$$r_t = \sigma_{sig}\left(W_i x_t + U_i h_{t-1} + b_i\right) \tag{7}$$

$$z_t = \sigma_{sig}\left(W_o x_t + U_o h_{t-1} + b_o\right) \tag{8}$$

$$h_t = z_t * h_{t-1} + \left(1 - z_t\right) * \sigma_{tanh}(W_c x_t + U_c\left(r_t * h_{t-1}\right) + b_{ct} \tag{9}$$

r_t is a reset gate which functions similarly to reset gate in an LSTM unit, while z_t is a single gate which combines the roles of output and input LSTM gates. The model requires six weight matrices (W_r, U_r, W_z, U_z, W_h, U_h) and three bias vectors (b_r, b_z, b_h).

4 Evaluation of RNN in Music Classification

The goal of our experiments is to compare the performance of GRU and LSTM in music audio classification, using low-level representation of music files. We want to assess whether music audio classification benefits from using gated architectures, how the performance of LSTM compares to GRU, and possibly identify additional issues specific to training recurrent networks on music data.

4.1 Datasets Description

For our experiments, we choose four benchmark datasets. The summary of contents is presented in Table 1. Music files in all datasets were cut to 30 s long.

Table 1. Contents of datasets

Dataset	Classes	Label type	Files
GTZAN	10	Genre	1000
Emotify	4	Genre	400
Ballroom	8	Dance Style	698
LastFM	4	Emotion	2904

First, we evaluate our networks on GTZAN. The dataset consists of 1000 music files in 10 genres (100 files each). While due to dataset imperfections the results on this set cannot be considered a good representation of the ability to classify actual musical genres (the faults or GTZAN are explained in Sect. 2.1) we can still demonstrate the behavior of RNN during learning and look for common patterns in the learning process between GTZAN and other datasets. However, we employ an additional genre dataset in order to have a more meaningful representation of our ability to recognize genres.

The second music genre recognition dataset is based on an emotion recognition dataset Emotify [19]. While it was originally meant for the task of predicting induced emotion, it provides a good variety of music in high quality. There are 400 excerpts evenly distributed between four genres: classical, rock, pop and electronic. The ground truth genres were given by Magnatune, a record label which supplied the music files.

The Ballroom dataset [20] was created for ISMIR 2004 data mining challenge (tempo recognition task) and is annotated with labels corresponding to eight ballroom dance styles: Cha Cha – 111 files, Jive – 60, Quickstep – 82, Samba – 98, Rumba – 86, Tango – 86, English Waltz – 110 and Viennese Waltz – 65. This is an unusual classification task, in which the main factor differentiating between dance styles is rhythm and tempo, which do not change significantly over the duration of the file. The music files are of a particularly low quality.

The fourth dataset, which we use as an example of an emotion recognition task, is the LastFM data mentioned in Sect. 2.1. Four classes corresponding to Valence-Arousal plane quadrants are: angry – 639 files, happy – 754, relaxed – 749 and sad – 763. The dataset was annotated based on popular tags from a social network, therefore its labels are not well-defined categories. This causes the classification accuracy achievable on this set to be rather low, with best known results being close to 55%.

4.2 Common Conditions of the Experiments

All neural networks in our experiments use a single recurrent layer so that the potential issues related to multilayer deep learning architectures can be omitted. A single non-recurrent layer on top of the recurrent network is used for classification. The non-recurrent layer uses a linear activation function, takes average of the recurrent layers' outputs as an input and has a single output for each class. Our implementation uses Theano [21] python library to compute gradients and parallelize the computation on GPU.

As a loss function, we use mean squared error (MSE) between ground truth labels and predictions. In preliminary experiments, we also tested a log-loss training regime and a softmax layer, but found no improvement from using either.

The size of the recurrent layer was set to 50. Since we are interested in the ability of RNN to work with low-level features, the input to the layer is a spectrogram of a music file. Spectrogram frames are 50 ms with 25 ms overlap, Mel scale is used for frequency and log scale for power. In order to shorten the length of input sequences below 1000 vectors, we build input vectors from two consecutive spectrogram frames. We also append the change from the previous frame for both frames in the input vector. Denoting concatenation as x|y we can define the i-th vector in training sequence as:

$$x_i = f_{2i-1} | f_{2i} \left| (f_{2i-2} - f_{2i-1}) \right| (f_{2i} - f_{2i-1}) \tag{10}$$

where f_i denotes i-th frame of the spectrogram.

The network is initialized with weights drawn from a Gaussian distribution with small values (mean 0, standard deviation 0.01) and trained using Adagrad optimizer with initial learning rate set to 0.05. We train for 150 epochs while monitoring the classification accuracy in order to show the speed of convergence and best achievable accuracy. We report the accuracy on both training and test dataset so that potential overfitting can be clearly visible. All reported results are averages from 10-fold cross-validation.

4.3 Basic Comparison of RNN Architectures

The first experiment compares the performance of gated architectures with a standard recurrent network. Results are presented in Figs. 1, 2, 3 and 4. We can see that both gated architectures outperform standard RNN. However, the training set performance is rather low. This indicates that either the model itself is incapable of learning the assigned tasks, or it stops in a local minimum.

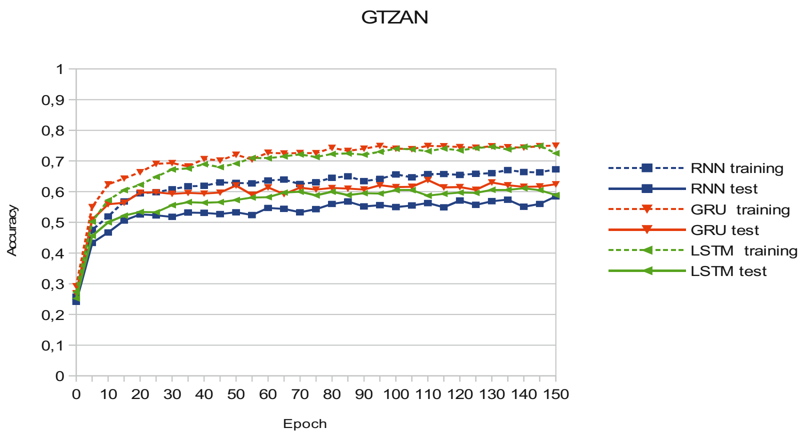


Fig. 1. Results of genre recognition on GTZAN dataset

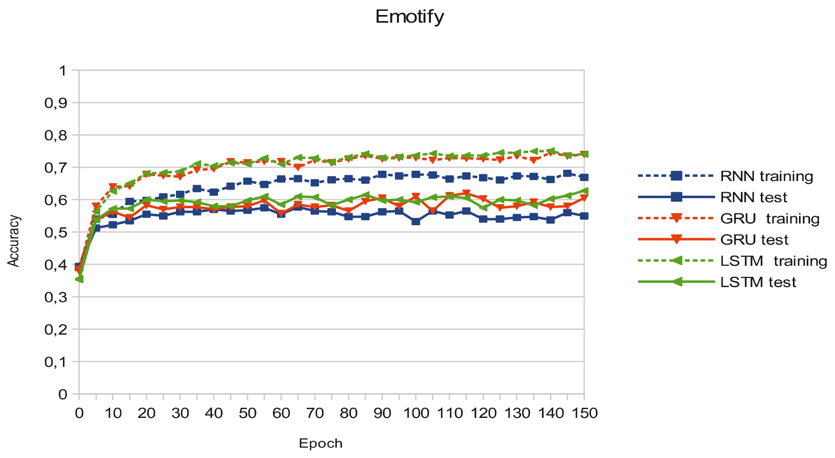


Fig. 2. Results of genre recognition on Emotify dataset

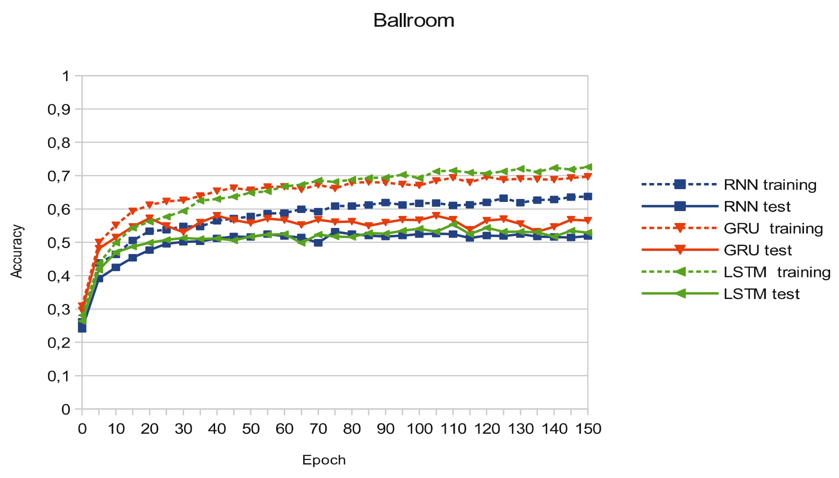


Fig. 3. Results of dance style recognition on Ballroom dataset

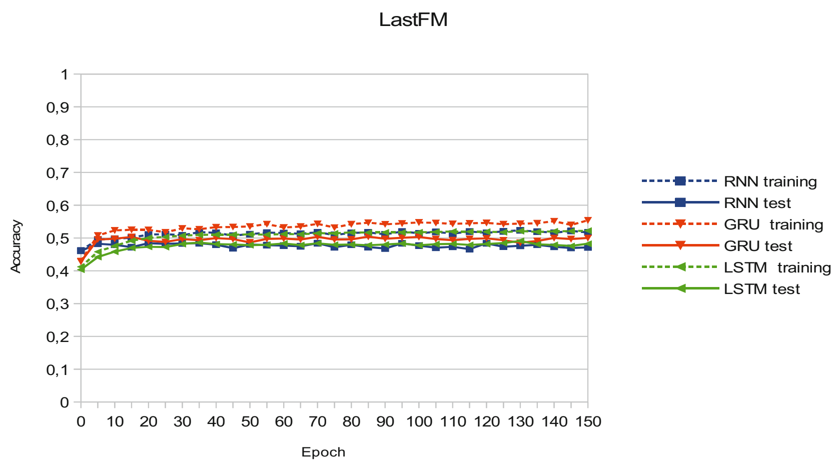


Fig. 4. Results of emotion recognition of LastFM dataset

On GTZAN dataset (Fig. 1), GRU converges faster on training data and outperforms LSTM on test data. On Emotify (Fig. 2), both architectures achieve similar results; both perform better than standard RNN. On Ballroom dataset (Fig. 3), we can see a different result. LSTM achieves better training set accuracy than GRU, while on test set it barely outperforms a standard RNN. This indicates that LSTM, in addition to aforementioned low training set accuracy, suffers from weak generalization. Finally, on LastFM set (Fig. 4) classification results are poor, however we can note that GRU outperforms standard RNN while LSTM does not.

Given low training set accuracy, a conclusion regarding the difference between gated unit types drawn from this experiment may be misleading. As we can clearly see, a standard approach using learning on full sequences cannot overcome bad local minima. We test whether it can be improved in the next experiment

4.4 Evaluation of Training on Random Short Excerpts

As one of the properties of audio music files we expected to influence the results was the length of input vector sequences, we want to evaluate training on excerpts of shorter length. We train the network on short excerpts of music files, selecting a subsequence corresponding to a 2.5-second fragment (100 spectrogram frames) from each song in the training set. The starting point of the subsequence is selected randomly (uniform distribution) in each epoch. During evaluation, the network processes full sequences. The results are presented in Table 2.

Table 2. Classification accuracy achieved by training on short excerpts randomly selected in each epoch compared to training on full songs (evaluated on full songs in both cases)

	LSTM				GRU			
	Training		Test		Training		Test	
	Full	Short	Full	Short	Full	Short	Full	Short
GTZAN	0.75	0.91	0.58	0.74	0.75	**0.92**	0.62	**0.75**
Emotify	0.74	**0.99**	0.63	0.67	0.74	0.97	0.60	**0.68**
Ballroom	0.73	**0.96**	0.53	**0.79**	0.69	**0.96**	0.56	**0.79**
LastFM	0.52	**0.54**	0.48	0.52	0.55	0.72	0.50	**0.54**

We can see an advantage of using the random subsequence approach over training on full sequences. On all datasets, performance on both test and training set improves after training on 2.5 s excerpts instead of full sequences. Performance on the training set is significantly better, indicating the ability to overcome local minima during the search.

Looking at the results on specific datasets, the most notable improvement is seen in the Ballroom data. The accuracy achieved on the test set using random subsequences approach is better than the accuracy on training set when training on full sequences. On the Emotify dataset, LSTM with random subsequences approach achieves nearly 100% fit to the training set. However, the test performance is not better than that achieved with GRU. In fact, despite achieving the best training performance on 3 out of 4 datasets, LSTM is still outperformed by GRU on the test set in every case. This is another indicator of the generalization issue, which we could notice on the Ballroom dataset in the first experiment.

5 Conclusions and Future Work

We have presented and evaluated an RNN-based approach to music file classification and compared the classification performance of two popular gated RNN architectures. While the results achieved using training on full sequences (Figs. 1, 2, 3 and 4) are poor, we proposed to train the network on random subsequences of sequences in the training set and demonstrated (Table 2) that this approach improves the results. Our results cannot be explained by a regularizing effect of randomness introduced to training process (similarly to e.g. adding noise to the input data). Unlike in regularization, the improvement concerns both training and test sets. It appears that the main problem with training on full sequences is that of averaging hidden layer outputs over a very long sequence, an issue which was not apparent in existing evaluations of LSTM and GRU on other types of data.

The comparison between LSTM and GRU demonstrates GRU achieves better results on test sets. However, the fit to training set is better for LSTM, indicating a generalization problem. A detailed study of regularization techniques in recurrent networks could deliver a definitive answer as to whether LSTM can outperform GRU.

Where a comparison in literature is available, the results can be described as promising. 54% accuracy in emotion recognition on LastFM songs is in line with the best known results on the dataset. 75% accuracy on GTZAN is lower than the best known results, although some of the best performing approaches in GTZAN are known to be inflated by the faults of the dataset and multiple results close to 90% accuracy have been challenged as non-replicable. In light of this fact, we find 75% accuracy promising enough to warrant more research. Future research should focus on the classification of long sequences. While training on random subsequences has shown to be effective, an attempt should be made to improve the learning process so that training on full sequences can achieve good results.

References

1. Sundermeyer, M., Schlüter R., Ney, H.: LSTM Neural Networks for Language Modeling. Interspeech *(*2012)
2. Graves, A., Santiago, F., Schmidhuber, J.: Bidirectional LSTM networks for improved phoneme classification and recognition. In: Artificial Neural Networks: Formal Models and Their Applications–ICANN 2005, pp. 753–753 (2015)
3. Schmidhuber, J., Hochreiter, S.: Long short-term memory. Neural Comput. **9**, 1735–1780 (1997)
4. Zhou, G., et al.: Minimal gated unit for recurrent neural networks. Int. J. Autom. Comput. **13**(3), 226–234 (2016)
5. Fu, Z., et al.: A survey of audio-based music classification and annotation. IEEE Trans. Multimedia **13**(2), 303–319 (2011)
6. Sturm, B.: The GTZAN dataset: Its contents, its faults, their effects on evaluation, and its future use (2013). arXiv preprint, arXiv:1306.1461
7. Tzanetakis, G., Cook, P.: Musical genre classification of audio signals. IEEE Trans. Speech Audio Process. **10**(5), 293–302 (2002)
8. Sturm, B.: An analysis of the GTZAN music genre dataset. In: Proceedings of the Second International ACM Workshop on Music Information Retrieval with User-Centered and Multimodal Strategies. ACM (2012)
9. Kim, Y.E., et al.: Music emotion recognition: a state of the art review. In: Proceedings of the 11th International Conference on Music Information Retrieval (2010)
10. Yang, Y., Chen, H.H.: Machine recognition of music emotion: a review. ACM Trans. Intell. Syst. Technol. **3**(3), 40 (2012)

11. Song, Y., Dixon, S., Pearce, M.: Evaluation of musical features for emotion classification. In: Proceedings of the 13th International Conference on Music Information Retrieval (2012)
12. Hamel, P., Wood, S., Eck, D.: Automatic identification of instrument classes in polyphonic and polyinstrument audio. In: Proceedings of the 10th International Conference on Music Information Retrieval, Kobe, Japan (2009)
13. Abeßer, J., Dittmar, C., Schuller, G.: Automatic recognition and parametrization of frequency modulation techniques in bass guitar recordings. In: Audio Engineering Society Conference: 42nd International Conference: Semantic Audio. Audio Engineering Society (2011)
14. Chung, J., et al.: Empirical evaluation of gated recurrent neural networks on sequence modeling (2014). arXiv preprint, arXiv:1412.3555
15. Greff, K., et al.: LSTM: A search space odyssey. IEEE Trans. Neural Netw. Learn. Syst. (2016)
16. Jozefowicz, R., Zaremba, W., Sutskever I.: An empirical exploration of recurrent network architectures. In: Proceedings of the 32nd International Conference on Machine Learning (2015)
17. Goller, C., Küchler, A.: Learning task-dependent distributed representations by backpropagation through structure. Neural Networks (1996)
18. Chung, J., Gulcehre, C., Cho, K.H., Bengio, Y.: Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling (2014)
19. Aljanaki, A., Wiering, F., Veltkamp, R.: Collecting annotations for induced musical emotion via online game with a purpose Emotify. Technical report Series 2014. UU-CS-2014-015 (2014)
20. Seyerlehner, K., Widmer, G., Schnitzer, D.: From rhythm patterns to perceived tempo. In: Proceedings of the 8th International Conference on Music Information Retrieval (2007)
21. Theano Development Team: "Theano: A Python framework for fast computation of mathematical expressions"

Neuro-Fuzzy System for Medium-Term Electric Energy Demand Forecasting

Paweł Pełka(✉) and Grzegorz Dudek

Department of Electrical Engineering,
Czestochowa University of Technology, Czestochowa, Poland
pavelle50@gmail.com, dudek@el.pcz.czest.pl

Abstract. Medium-term electric energy demand forecasting plays an important role in power system planning and operation as well as for negotiation forward contracts. This paper proposes a solution to medium-term energy demand forecasting that covers definition of input and output variables and the forecasting model based on a neuro-fuzzy system. As predictors patterns of the yearly periods of the time series are defined, which unify input data and filter out the trend. Output variable is encoded in tree ways using coding variables describing the process. For prediction of coding variables, which are necessary for postprocessing, ARIMA and exponential smoothing models are applied. The simplified relationship between preprocessed input and output variables is modeled using Adaptive-Network-Based Fuzzy Inference System. As an illustration, we apply the proposed time series forecasting methodology to historical monthly energy demand data in four European countries and compare its performance to that of alternative models such as ARIMA, exponential smoothing and kernel regression. The results are encouraging and confirm the high accuracy of the model and its competitiveness compared to other forecasting models.

Keywords: Neuro-Fuzzy systems · Medium-term load forecasting · Pattern-based forecasting

1 Introduction

Accurate medium-term electric energy demand forecasting plays an essential role for electric power system planning and operation, and offers significant benefits for companies operating in a regulated and deregulated energy markets. Generally, it is used to optimize energy production and transmission and improve power system reliability. It is necessary for scheduling and coordinate maintenance and production across a power system, negotiation fuel purchases for power stations, and optimization of renewable energy sources, such as wind farms.

Characteristic feature of the electricity demand time series is the yearly seasonality corresponding to climatic factors and weather variations. Also a trend is observed following the economic and technological development of a country, and random component disturbing the time series. These features should be included in the forecasting process to increase the prediction accuracy.

The medium-term load forecasting (MTLF) methods can be categorized into two general groups [1]. The first group, referred to as the conditional modeling approach, focuses on economic analysis, management and long term planning and forecasting of energy load and energy policies. It takes into account changes in socioeconomic conditions which impact energy demands. Input variables include historical load data and weather factors as well as economic indicators and electrical infrastructure measures. Efforts of researchers in this field are focused on definition of optimal set of input variables and construction of appropriate forecasting models. An example of a model of this type can be found in [2], where macroeconomic indicators, such as the consumer price index, average salary earning and currency exchange rate are taken into account as inputs.

The second group, referred to as the autonomous modeling approach, requires a smaller set of input information to forecast future electricity demand. Primarily past loads and weather variables. This approach is more suited for stable economies. The forecasting models used in this group include classical methods such as ARIMA or linear regression [3] as well as computational intelligence methods, such as neural networks [4, 5] and support vector machines [6].

The forecasting model proposed in this work can be classified to autonomous modeling approach. It uses neuro-fuzzy network which works on preprocessed data. Inputs are defined as patterns of yearly fragments of the demand time series, which are normalized version of demand vectors. Outputs are encoded demands. The proposed way of time series preprocessing unifies data and filters out a trend.

The remaining sections of the paper are organized as follows. In Sect. 2, time series representation is described. Section 3 presents in detail a neuro-fuzzy forecasting model. In Sect. 4, we evaluate the performance of the model in monthly electricity demand forecasting using real-world data. Finally, Sect. 5 is a summary of our conclusions.

2 Time Series Representation

Let us consider the task of prediction of the monthly electricity demand with horizon τ. The predicted time series point is $E_{i+\tau}$. As predictors we use preprocessed n points preceding the forecasted point, i.e. the time series fragment $X_i = \{E_{i-n+1}, E_{i-n+2}, \ldots, E_i\}$. This fragment is represented by input pattern $\mathbf{x}_i = [x_{i,1}\ x_{i,2}\ \ldots\ x_{i,n}]^T$. The components of this vector are defined as follows [7]:

$$x_{i,t} = \frac{E_{i-n+t} - \bar{E}_i}{D_i} \tag{1}$$

where: $t = 1, 2, \ldots, n$, $\bar{E}_i$ is the mean value of the points in sequence X_i, and $D_i = \sqrt{\sum_{j=1}^{n} (E_{i-n+j} - \bar{E}_i)^2}$ is a measure of their dispersion.

Note, that the x-pattern defined using the above equation is a normalized vector $[E_{i-n+1}\ E_{i-n+2}\ \ldots\ E_i]^T$. It has the unity length and the mean value equal to zero. Moreover, x-patterns representing different fragments have the same variance. Thus, the time series, which is nonstationary, is represented by unified patterns, having the same mean and variance. The trend is filtered out. When $n = 12$ the x-pattern carries information about the shape of the yearly cycle.

The output variable, $E_{i+\tau}$, is encoded in three ways. In the first approach (C1) the forecasted value is encoded as follows:

$$y_i = \frac{E_{i+\tau} - \bar{E}_i}{D_i} \tag{2}$$

where $\bar{E}_i$ and D_i are determined from the sequence X_i.

Note, that $\bar{E}_i$ and D_i are known at the moment of making the forecast (moment i) and can be used for calculation the forecast of demand based on the forecast of y_i returned by the forecasting model. We use for this the transformed Eq. (2):

$$\widehat{E}_{i+\tau} = \widehat{y}_i D_i + \bar{E}_i \tag{3}$$

In the second approach (C2) $\bar{E}_i$ and D_i are determined from the annual period following the period X_i, i.e. the period including time series fragment $\{E_{i+1}, E_{i+2}, \ldots, E_{i+12}\}$. This approach is used for forecast horizon $\tau \in \{1, 2, \ldots, 12\}$. Note, that in this case coding variables $\bar{E}_i$ and D_i are not available at the time of making the forecast. Thus, they should be forecasted. In the experimental part of the work we use ARIMA and exponential smoothing (ES) for forecasting the coding variables.

In the third approach (C3), which is used only for one-step ahead forecasts ($\tau = 1$), the coding variables $\bar{E}_i$ and D_i are determined from the annual period including time series fragment $\{E_{i-n+2}, E_{i-n+3}, \ldots, E_{i+1}\}$. In this case coding variables cannot be calculated from time series elements because the value of E_{i+1} is not known. Thus, $\bar{E}_i$ and D_i should be predicted. Just like in the case of C2, we use for this ARIMA and ES.

3 Neuro-Fuzzy Forecasting System

The proposed forecasting model is based on Adaptive-Network-Based Fuzzy Inference System (ANFIS) developed by Jang [8]. This is a multi-input, single-output quasi-nonlinear model consisting of a set of linguistic if-then rules. Its architecture is functionally equivalent to a Sugeno type fuzzy rule base. In Fig. 1 ANFIS architecture in application to the energy demand forecasting is shown. Squares in this figure indicate adaptive nodes, whereas circles indicate fixed nodes (without parameters).

The functions of ANFIS nodes in subsequent layers are described below.

Layer 1. A node represents a membership function. In our case this is a Gaussian function of the form:

$$\mu_{A_k^m}(x_k) = \exp\left[-\left(\frac{x_k - c_k^m}{\sigma_k^m}\right)^2\right] \tag{4}$$

where $m = 1, 2, \ldots, M$ is the fuzzy set number, $k = 1, 2, \ldots, 12$ is the x-pattern component number, A_k^m is the fuzzy set describing linguistically the input component x_k, c_k^m and σ_k^m are premise parameters: center and spread, respectively.

An output of the node expresses a membership degree of x_k in the fuzzy set A_k^m. The number of nodes is determined by the number of linguistic labels M and the x-pattern length.

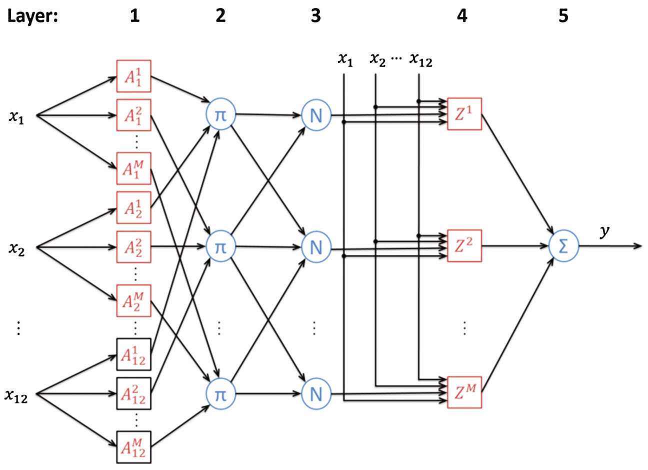


Fig. 1. ANFIS architecture.

Layer 2. A node expresses a firing strength of the *m*th rule. It is calculated as the product t-norm:

$$\alpha^m = \prod_{k=1}^{12} \mu_{A_k^m}(x_k) \tag{5}$$

The firing strength of the *m*th rule depends on the membership degree of each x-pattern component in the relevant fuzzy set. It has the highest value if the x-pattern components coincide with the centers of the membership functions.

Layer 3. A node expresses a normalized firing strength for the *m*th rule:

$$\bar{\alpha}^m = \frac{\alpha^m}{\sum_{j=1}^{M} \alpha^j} \tag{6}$$

Layer 4. A node expresses a conclusion of the *m*th rule. Conclusion is determined using the Takagi-Sugeno-Kang method, where the output membership functions are either linear or constant (first or zeroth order Sugeno-type systems). Each rule weights its output level by the firing strength of the rule. The node function for the first order system is of the form:

$$z^m = \bar{\alpha}^m \left(\sum_{k=1}^{12} a_k^m x_k + b^m \right) \tag{7}$$

where a_k^m and b^m are consequent parameters.

Layer 5. A node computes the overall system output as the sum of all incoming signals:

$$y = \sum_{m=1}^{M} z^m = \frac{\sum_{m=1}^{M} \alpha^m \left(\sum_{k=1}^{12} a_k^m x_k + b^m \right)}{\sum_{j=1}^{M} \alpha^j} \tag{8}$$

Note, that the output of each rule is a linear combination of inputs and the final output of the system is the weighted average of all rule outputs. Because the membership functions are nonlinear in our case, the weights (firing strengths) are dependent on inputs nonlinearly, and the final output is nonlinear.

I our case the rule base is of the form:

$$\begin{aligned}
&\text{If } x_1 \text{ is } A_1^1 \text{ and}\ldots\text{and } x_{12} \text{ is } A_{12}^1 \text{ then } z^1 = \sum_{k=1}^{12} a_k^1 x_k + b^1 \\
&\text{If } x_1 \text{ is } A_1^2 \text{ and}\ldots\text{and } x_{12} \text{ is } A_{12}^2 \text{ then } z^2 = \sum_{k=1}^{12} a_k^2 x_k + b^2 \\
&\ldots \\
&\text{If } x_1 \text{ is } A_1^M \text{ and}\ldots\text{and } x_{12} \text{ is } A_{12}^M \text{ then } z^M = \sum_{k=1}^{12} a_k^M x_k + b^M
\end{aligned} \tag{9}$$

The premise part of each rule defines a fuzzy region for the linear model included in the consequence part. The inference mechanism interpolates smoothly between each local model to provide a global model.

Before the training, an initialization of ANFIS is required. Initial positions of the membership functions in the premise parts of rules are determined using fuzzy c-means clustering on the input data. The number of clusters corresponding to the numbers of rules *M* is selected in leave-one-out cross-validation procedure. This parameter decides about the bias–variance tradeoff. Increasing *M* we increase the model variance and decrease its bias.

For ANFIS training, i.e. estimation of the premise and consequent parameters, a hybrid learning algorithm is applied. It uses a combination of the least-squares and backpropagation gradient descent methods to model the training data set. The error measure minimized during training is defined by the sum of the squared difference between actual and desired outputs.

4 Application Example

In order to assess the performance of the proposed forecasting model to obtain generalized conclusions, we use it to forecast monthly electricity demand for four European countries: Poland (PL), Germany (DE), Spain (ES) and France (FR). The data used for the experiments were retrieved from the ENTSO-E repository (www.entsoe.eu). The datasets contain monthly electricity demand from the period 1998-2015 for PL and 1991-2015 for other countries. The forecasts are made for 2015, using data from previous years as training data. The only model parameter is *M* (number of rules in ANFIS). It was selected for each ANFIS model from the range 2-13 in the leave-one-out cross-validation.

The forecasts were generated in two procedures. In the first procedure (A) the forecasts for successive 12 months of 2015 are generated by 12 ANFIS models. Each model gets the same input pattern representing time series fragment from January to December 2014, and produces a forecast for *k*th month of 2015 (k = 1, 2,…, 12). Thus, the forecast horizon for the model for January 2015 is $\tau = 1$, for the model for February 2015 is $\tau = 2$, etc. The output variable is encoded using C1 or C2 approach. In the latter case coding variables $\bar{E}_i$ and D_i for 2015 are predicted using ARIMA and ES on the basis of their historical values, i.e. values determined for 12 months of the successive years. The results of forecasting in Fig. 2 are shown.

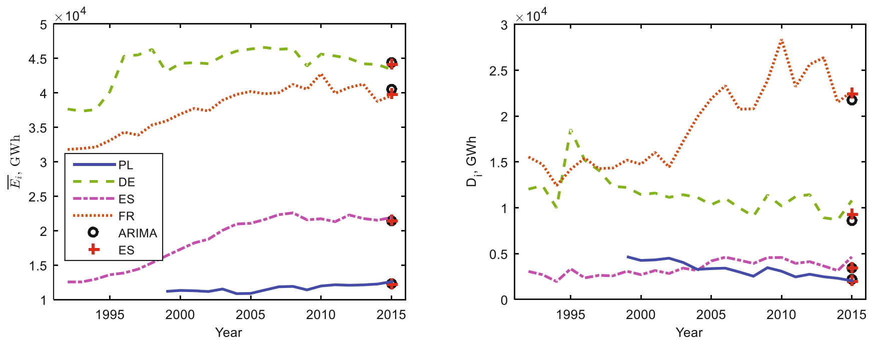


Fig. 2. Forecasts of coding variables.

In the second procedure of forecasting (B) one-step ahead forecasts are generated ($\tau = 1$) for successive months of 2015. The input patterns for the models represent 12 preceding months, i.e. the model for January 2015 gets input pattern representing time series fragment from January to December 2014, the model for February 2015 gets input pattern representing time series fragment from February 2014 to January 2015, etc. The output variable is encoded using C1 or C3 approach. The latter case needs the coding variables $\bar{E}_i$ and D_i to be predicted. We use for this ARIMA and ES, as in the A procedure.

The real and forecasted values of monthly demand are presented in Figs. 3 and 4, and errors for each month of the test period in Figs. 5 and 6. Forecast errors for validation and test samples in Tables 1 and 2 are presented. In these tables results of comparative models are also shown: ARIMA, ES and Nadaraya-Watson estimator (N-WE) [7]. As can be seen from the figures and tables, it is hard to select the best model variant. For PL data the lowest errors gives variant C1 for both A and B forecasting procedures, and the worst variant is C2-ES. C1 is also the best for DE, variant B. For three out of eight cases the variants using ES, i.e. C2-ES and C3-ES, turned out to be the most accurate among the proposed ones. And variants using ARIMA were the best in two cases. When comparing errors of all models, it should be noted that the classical ES model outperformed all other models in five of eight cases.

Comparing results for A and B procedures we can conclude that variant B which generates one-step ahead forecasts, usually provides better results than variant A. An exception is FR data, where higher errors in variant B are observed. Due to significant contribution of the random component in the time series the errors for successive months are very varied. Different level of heteroscedasticity in time series for different countries and also occurrence of nonlinear trend in some of them cause deterioration in the model accuracy.

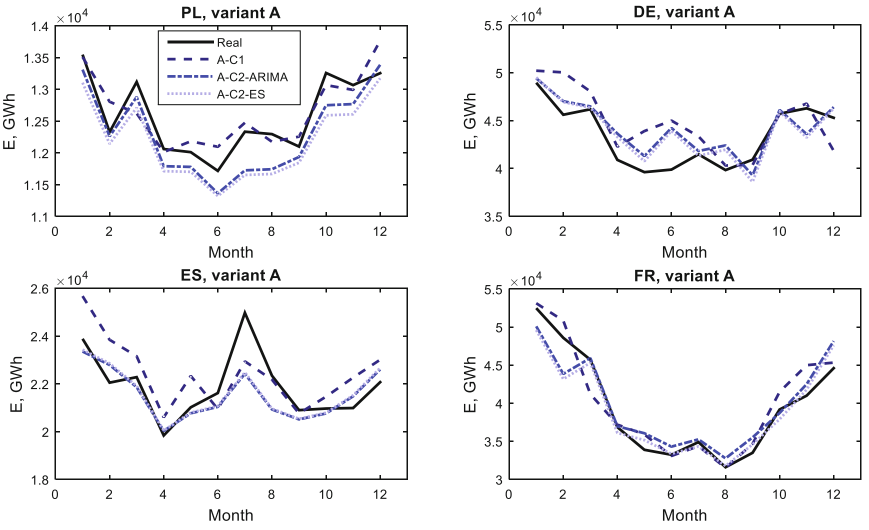


Fig. 3. Real and forecasted monthly demand for 2015, variant A.

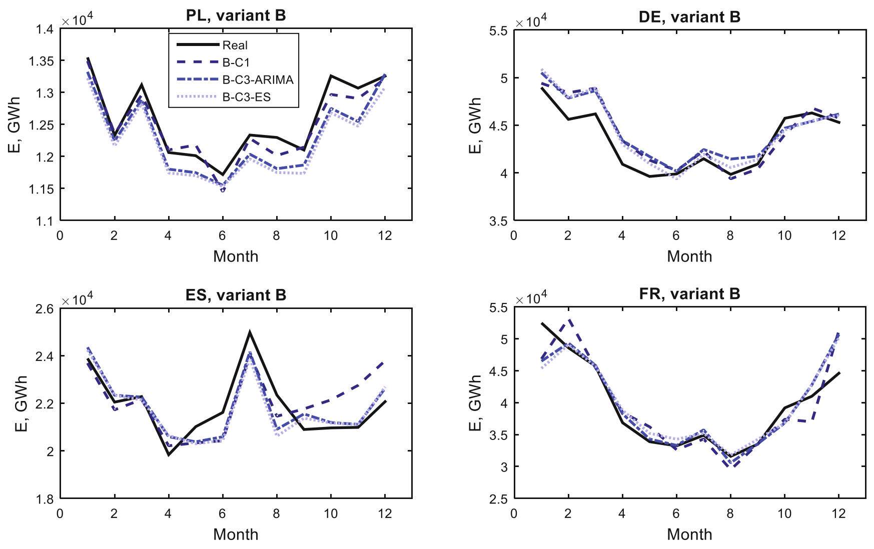


Fig. 4. Real and forecasted monthly demand for 2015, variant B.

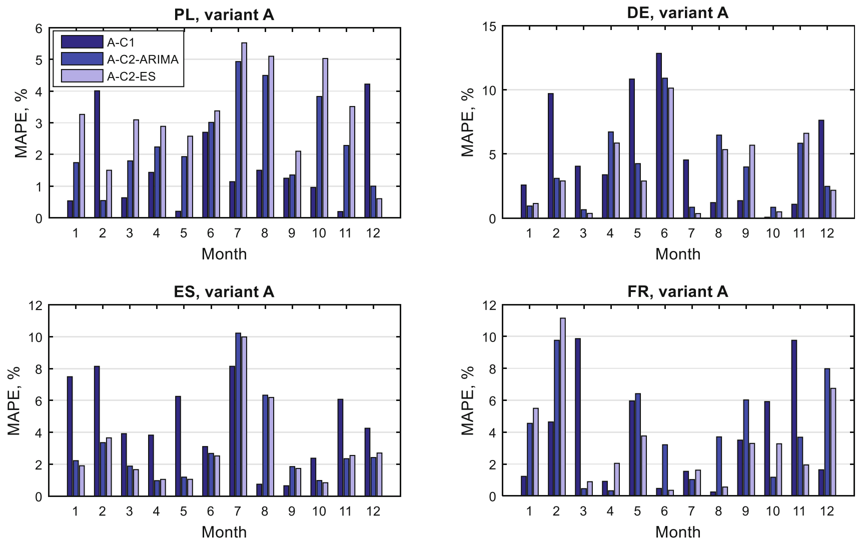


Fig. 5. Errors for consecutive months of 2015, variant A.

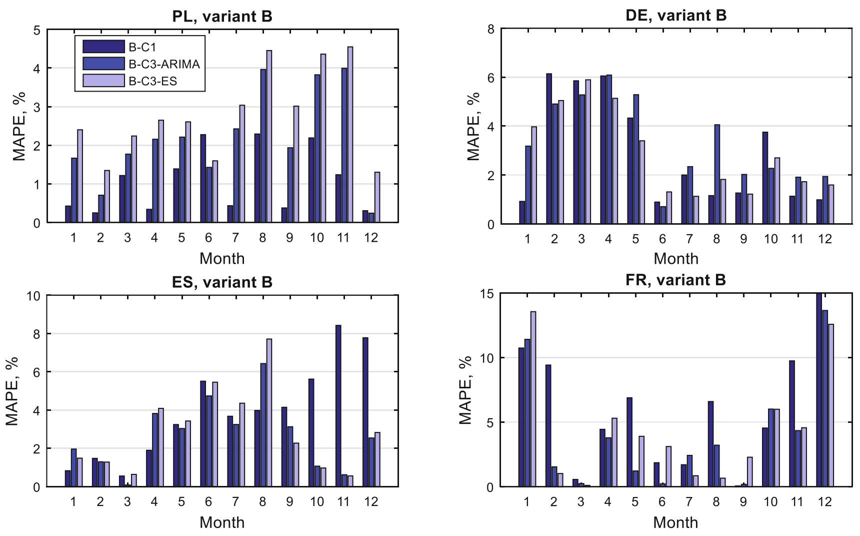


Fig. 6. Errors for consecutive months of 2015, variant B.

Table 1. Forecast errors, variant A.

	PL		DE		ES		FR	
	$MAPE_{val}$	$MAPE_{tst}$	$MAPE_{val}$	$MAPE_{tst}$	$MAPE_{val}$	$MAPE_{tst}$	$MAPE_{val}$	$MAPE_{tst}$
A-C1	2.27	1.57	2.96	4.93	2.63	4.58	3.57	3.81
A-C2-ARIMA	1.57	2.43	1.82	3.92	2.00	3.04	2.64	4.02
A-C2-ES	1.57	3.21	1.82	3.66	2.00	2.99	2.64	3.43
ARIMA	–	2.08	–	2.54	–	2.67	–	4.02
ES	–	1.92	–	2.32	–	2.17	–	3.02
N-WE	–	2.03	–	3.12	–	2.08	–	3.56

Table 2. Forecast errors, variant B.

	PL		DE		ES		FR	
	$MAPE_{val}$	$MAPE_{tst}$	$MAPE_{val}$	$MAPE_{tst}$	$MAPE_{val}$	$MAPE_{tst}$	$MAPE_{val}$	$MAPE_{tst}$
B-C1	2.04	1.06	2.70	2.87	2.33	3.92	3.14	5.95
B-C3-ARIMA	1.67	2.19	2.32	3.33	1.92	2.66	2.64	4.01
B-C3-ES	1.67	2.80	2.32	2.91	1.92	2.92	2.64	4.49
ARIMA	–	2.02	–	2.56	–	2.18	–	3.91
ES	–	1.92	–	2.32	–	2.16	–	2.98
N-WE	–	1.35	–	2.72	–	3.42	–	3.99

5 Conclusion

This work presents an ANFIS model which is used for medium-term electric demand forecasting. The model works on preprocessed time series fragments - patterns of yearly cycles. The patterns unify data and reduce nonstationarity. The novelty of this work is that output variable is encoded in three ways using coding variables determined from history or forecasted using ARIMA or exponential smoothing. The advantage of the ANFIS is that despite the complex structure, there is only one parameter to be tuned – the number of rules.

In the light of the experimental study, it can be concluded that neuro-fuzzy inference models have been proven to be useful in medium-term load forecasting. Their accuracy depend on time series features. For PL data set ANFIS model in its basic variant (C1) provided the best results. But for other data sets other models generated better results. In the future work, we are going to test the ANFIS forecasting model thoroughly in medium-term electric demand forecasting for other European countries.

References

1. Ghiassi, M., Zimbra, D.K., Saidane, H.: Medium term system load forecasting with a dynamic artificial neural network model. Electr. Power Syst. Res. **76**, 302–316 (2006)
2. Gavrilas, M., Ciutea, I., Tanasa, C.: Medium-term load forecasting with artificial neural network models. In: 16th International Conference and Exhibition on Electricity Distribution (2001)
3. Hor, C.L., Watson, S., Majithia, S.: Analyzing the impact of weather variables on monthly electricity demand. IEEE Trans. Power Syst. **20**, 2078–2085 (2005)
4. González-Romera, E., Jaramillo-Morán, M., Carmona-Fernández, D.: Monthly electric energy demand forecasting based on trend extraction. IEEE Trans. Power Syst. **21**, 1946–1953 (2006)
5. Chen, J.F., Lo, S.K., Do, Q.H.: Forecasting monthly electricity demands: an application of neural networks trained by heuristic algorithms. Information **8**, 31 (2017)
6. Hu, Z., Bao, Y., Chiong, R., Xiong, T.: Mid-term interval load forecasting using multi-output support vector regression with a memetic algorithm for feature selection. Energy **84**, 419–431 (2015)
7. Dudek, G., Pełka, P.: Medium-term electric energy demand forecasting using Nadaraya-Watson estimator. In: 18th International Scientific Conference on Electric Power Engineering (EPE), pp. 1–6, (2017)
8. Jang, J.S.R.: ANFIS: adaptive-network-based fuzzy inference system. IEEE Trans. Syst. Man Cybern. **23**(3), 665–685 (1993)

An Evolutionary Optimization Method Based on Scalarization for Multi-objective Problems

Marcin Studniarski[1(✉)], Radhwan Al-Jawadi[2,3], and Aisha Younus[4]

[1] Faculty of Mathematics and Computer Science, University of Łódź, Banacha 22, 90-238 Łódź, Poland
marstud@math.uni.lodz.pl

[2] Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, Warsaw, Poland
radwanyousif@mimuw.edu.pl, radwanyousif@yahoo.com

[3] Technical College of Mosul, Mosul, Iraq

[4] MSc Computer Science, Faculty of Mathematics and Computer Science, University of Łódź, Łódź, Poland
azeezzena74@yahoo.com

Abstract. In this paper, we perform some computational experiments on the new global scalarization method for multi-objective optimization problems. Its main idea is to construct, for a given multi-objective optimization problem, a global scalarization function whose values are non-negative real numbers. The points where the scalarization function attains the zero value are exactly weak Pareto stationary points for the original multi-objective problem.

We apply two different evolutionary algorithms to minimize the scalarization function; both of them are designed for solving scalar optimization problems. The first one is the classical Genetic Algorithm (GA). The second one is a new algorithm called Dissimilarity and Similarity of Chromosomes (DSC), which has been designed by the authors.

The computational results presented in this paper show that the DSC algorithm can find more minimizers of the scalarization function than the classical GA.

Keywords: Scalarization · Multi-objective optimization · Genetic algorithm · Dissimilarity and similarity of chromosomes algorithm

1 Introduction

Since 1960's, growing attention has been paid to evolutionary methods of optimization which intend to mimic the nature evolution fundamentals. This idea has influenced the design of optimization algorithms and stochastic searches. Instead of utilizing a single solution just like the classical methods, evolutionary methods are utilizing random solutions as base populations. To attain the optimal solutions, these base populations are updated in each iteration. In addition to that, evolutionary methods can be used to solve multi-objective optimization problems (MOO) by giving multiple solutions [1, 2].

One of the well-known evolutionary methods is the classical Genetic Algorithm (GA) described in Chap. 2 of [3]. It uses the roulette-wheel selection and two genetic operators: crossover and mutation. Unfortunately, the selection method used there is not suitable for MOO problems since it requires a single scalar-valued fitness function. Therefore, in order to be able to apply the classical GA to an MOO problem, one will need some scalarization procedure that converts the MOO problem to some single-objective (SOO) problem. Of course, there are also other evolutionary algorithms for solving MOO problems which do not depend on any scalarization, for example, NSGA-II [4].

Some typical scalarization methods are briefly reviewed in the next section. In Sects. 5 and 6, we describe the new global scalarization method introduced by Rahmo and Studniarski in [5]. In this paper, we consider only the special case where we have the usual positive cone in $\mathbb{R}^p$ instead of an arbitrary ordering cone. This simplifies the proof of the main statement (Proposition 3) which is included here for the reader's convenience. We construct a single scalarization function for an MOO problem which has the property that its global minimizers are exactly weak Pareto stationary points for the original multi-objective problem.

The aim of this paper is to present some numerical experiments on solving three simple MOO problems taken from the literature, by using global minimization of the scalarization function. We apply and compare two evolutionary algorithms: the classical GA (see the comments in Sect. 3) and a new algorithm called Dissimilarity and Similarity of Chromosomes (DSC) originally described in an earlier paper by the authors [6]. A short description of the DSC algorithm is repeated below in Sect. 4. Section 7 contains the presentation of the three selected examples of MOO problems. In Sect. 8, we discuss numerical results of applying the two mentioned evolutionary algorithms. Finally, Sect. 9 contains some conclusions and recommendations for a future work.

2 Scalarization and Optimality Conditions

In this section, we discuss scalarization of multi-objective optimization problems.

Scalarization is considered as one of possible methods for solving MOO problems. In this method, a MOO problem is transformed into a family of scalar optimization problems usually parameterized with a parameter vector. Scalarization methods should have the following two properties:

1. An optimal solution of each scalarized problem is efficient (properly efficient or weakly efficient), for the original MOO problem.
2. Every efficient solution of the MOO problem can be obtained as an optimal solution of an appropriate scalarized problem by choosing an appropriate parameter value.

The main schema of scalarization for an MOO problem is the following:

$$\min g(f_1(\mathrm{x}), \ldots, f_k(\mathrm{x})) \text{ subject to } x \in X, \tag{1}$$

where $f_1, \ldots, f_k$ are optimality criteria and g is a given scalar-valued function.

There are several typical methods for scalarization, like weighted sum minimization (see [7, 8]), ε-constraint method (see [8–10]), Tchebyshev scalarization (see [11]).

3 Comments on Genetic Algorithms

One of the most important optimization methods is the Genetic Algorithm (GA). It depends on guided random searches to obtain the optimal solution. It is inspired by the process of evolution appearing in nature, and involves simulation of the mating process between organisms of the same type, where several characteristics have been borrowed from genetics science, such as generation, parent, crossing and mutation. Then, these techniques are used to access the most appropriate solution [12].

The process of applying a GA to a given problem requires an appropriate coding of solutions. One of the most known problem coding methods is the binary representation [3, 13]. GAs have confirmed their ability to solve many different problems since several decades. This is because of their characteristics of inherent parallel processing, global perspective, and simplicity [12].

A GA can include some standard operations like: representation (binary or real coding), selection procedures (roulette wheel, ranking or tournament selection or elitism), crossover and mutation; see [13–15].

4 The DSC Algorithm [6]

An optimization problem that is considered in this section is as follows:

$$\text{minimize} f(x_1, \ldots, x_n) \text{ subject to}: \quad x_i \in [a_i, b_i],\ i = 1, \ldots, n \tag{2}$$

where f: $\mathbb{R}^n \to \mathbb{R}$ is a given function.

In the algorithm described below, a standard encoding is used for chromosomes as in [3]. Let M be a positive integer divisible by 8. The DSC algorithm works as follows:

1. Generate M chromosomes, each chromosome representing a point $(x_i), i = 1, \ldots n$.
2. Compute the values of the fitness function f for each chromosome in the population.
3. Sort the chromosomes according to the descending (for maximization) or ascending (for minimization) values of the fitness function.
4. Copy C times the first chromosome and put it randomly in C positions in the first half of the population replacing the original chromosomes, where C = M/8.
5. Compare pairs of chromosomes for the first half of the population to find dissimilarities and similarities. Check each two sequential chromosomes, that is the first and the second; the second and the third; and so on, by comparing the respective bits as follows:
   a. For the chromosomes in the first quarter of the population (from 1 to M/4): if the two bits are equal, put a star (*) in the second (following) chromosome; otherwise leave this bit without a change in the second chromosome. Then put randomly 0 or 1 in the bits with stars (*). Compare this new second chromosome with the third one and so on.
   b. For the chromosomes in the second quarter of the population (from M/4 + 1 to M/2): if the two bits are not equal, put a star (*) in the second (following) chromosome; otherwise leave this bit without a change in the second chromosome.

Then put randomly 0 or 1 in the bits with stars (*). Compare this new second chromosome with the third one and so on.

6. All chromosomes B created in step 5 replace the original chromosomes in the positions from 2 to M/2. Then we generate randomly chromosomes for the second half of population. These will replace the second half of the chromosomes (the positions from M/2 + 1 to M).
7. Go to step 2 and repeat until the stopping criterion is reached.

5 The Idea of New Scalarization

Let: $f = (f_1, \ldots, f_p) : \mathbb{R}^n \to \mathbb{R}^p$ be a locally Lipschitzian mapping. Suppose that we want to solve the following MOO problem:

$$\text{minimize } f(x) \quad \text{subject to } x \, \epsilon \mathbb{R}^n \tag{3}$$

We say that a point $\bar{x} \in \mathbb{R}^n$ is a weak Pareto minimizer for problem (3) if there is no $x \in \mathbb{R}^n$ such that:

$$f_i(x) < f_i(\bar{x}) \text{ for all } i \epsilon I := \{1, \ldots, p\}. \tag{4}$$

The following theorem on necessary conditions is true, which is a particular case of [16], Theorem 3.1. We denote by $\partial f_i(\bar{x})$ Clarke's generalized gradient of f_i at $\bar{x}$, see [17].

Theorem 1: If $\bar{x}$ is a weak Pareto minimizer for the problem (3), then there exists a vector $\lambda = (\lambda_1, \ldots, \lambda_p) \in \mathbb{R}^p$ such that:

$$\lambda_i \geq 0 (i \in I), \quad \sum\nolimits_{i \in I} \lambda_{i=1} \text{ and } 0 \in \sum\nolimits_{i \in I} \lambda_i \partial f_i(\bar{x}). \tag{5}$$

We say that a point $\bar{x} \in \mathbb{R}^n$ is a weak Pareto stationary point for problem (3) if there exists a vector $\lambda \in \mathbb{R}^p$ satisfying conditions (5). Using the convexity of the sets $\partial f_i(\bar{x})$, if it is easy to verify the following:

Proposition 2: A point $\bar{x} \in \mathbb{R}^n$ is a weak Pareto stationary point for problem (3) if and only if

$$0 \in \text{co} \cup_{i \in I} \partial f_i(\bar{x}), \tag{6}$$

where co denotes the convex hull.

For a nonempty subset A of $\mathbb{R}^n$, let $d(., A) : \mathbb{R}^n \to \mathbb{R}$ be the distance function of A, defined as follows:

$$d(x, A) := \inf \{\|x - a\| : a \in A\}, \tag{7}$$

where $\|.\|$ denotes the Euclidean norm. We introduce the following function $s : \mathbb{R}^n \to [0, +\infty)$:

$$s(x) := d(0, \text{co} \cup_{i \in I} \partial f_i(x)). \tag{8}$$

We will call s a scalarization function for problem (3) because it satisfies the following property:

Proposition 3: A point $\bar{x} \in \mathbb{R}^n$ is a weak Pareto stationary point for problem (3) if and only if $s(\bar{x}) = 0$.

Proof. if $\bar{x}$ is weak Pareto stationary point for (3), then by Proposition 2, $0 \in \text{co} \cup_{i \in I} \partial f_i(x)$, which gives $s(\bar{x}) = 0$. Conversely, suppose that $s(\bar{x}) = 0$. Since the sets $\partial f_i(\bar{x})$, $i \in I$, are compact in $\mathbb{R}^n$ (see [17], Proposition 2.1.2(a)), the set co $\cup_{i \in I} \partial f_i(\bar{x})$, is also compact; hence it is closed. Therefore, the equality $s(\bar{x}) = 0$ implies condition (6).

Now problem (3) can be replaced by the following SOP:

$$\text{minimize } s(x) \text{ subject to } x \in \mathbb{R}^n. \tag{9}$$

Because some weak Pareto stationary points might occur that are actually not weak Pareto minimizers for (3), so, problems (3) and (9) are not equivalent. However, some approximation of the set of solutions to (3) could be achieved by solving problem (9).

Unfortunately, the distance function (7) is not easy to be calculated. But with the differentiability assumptions for the two objectives, some representation of the scalarization function s can be found in terms of the gradients ∇f_2 and ∇f_1. This will be explained in the next section.

6 A Problem with Two Objectives

Let $p = 2$ and suppose that the mapping $f = (f_1, f_2)$ is continuously differentiable on $\mathbb{R}^n$. Denote by $\nabla f_i(x)$ the gradient of f_i at $x (i = 1, 2)$. Then the scalarization function has the form [5]:

$$s(x) = d(0, \text{co}\{\nabla f_1(x), \nabla f_2(x)\}). \tag{10}$$

For any point $x \in \mathbb{R}^n$, there are two possible cases: (i) $\nabla f_1(x) = \nabla f_2(x)$, then $s(x) = \|\nabla f_1(x)\| = \|\nabla f_2(x)\|$; (ii) $\nabla f_1(x) \neq \nabla f_2(x)$, then $s(x)$ is the distance from 0 to the line segment S joining the points $\nabla f_1(x)$ and $\nabla f_2(x)$.

Now considering the case (ii), the line L passing through $\nabla f_1(x)$ and $\nabla f_2(x)$ is parameterized as $\text{L(t)} = \text{b} + \text{ta}$, where $b := \nabla f_1(x)$ is a point on the line, and

$a := \nabla f_2(x) - \nabla f_1(x)$ is the line direction. The closest point on the line L to 0 is the projection of 0 onto L which is equal to:

$$q := b + t_0 a \text{ where } t_0 = -\frac{\langle a, b\rangle}{\langle a, a\rangle} = -\frac{\langle a, b\rangle}{\|a\|^2} \tag{11}$$

where $\langle .,. \rangle$ is the inner product in $\mathbb{R}^n$. Using the same parameterization, the line segment S can be represented as follows:

$$S = \{b + ta : 0 \le t \le 1\}. \tag{12}$$

It is shown in [5] that the function s can be described as follows:

$$s(x) = \begin{cases} \|b\| & \textit{if } t_0 \le 0, \\ \|b + t_0 a\| & \textit{if } 0 < t_0 < 1 \\ \|b + a\| & \textit{if } t_0 \ge 1. \end{cases} \tag{13}$$

7 Test Functions (Problems)

In this section, we present three test MOO problems described by three vector functions with different numbers of variables. In Sect. 8, we will apply the new scalarization method presented in Sects. 5 and 6 to these problems. This will involve minimizing the respective scalarization functions by using the GA and DSC algorithms. The results show that this method can be implemented with the GA, but it is more successful with the new optimization algorithm DSC, see Sect. 4. All the three problems are minimization problems. The functions are described as follows [2, 4]:

7.1 SCH Function

In this function, $f_1(x) = x^2, f_2(x) = (x-2)^2$, the number of variables is equal to 1; the interval for finding solutions is $x \in [-1000, 1000]$; the functions f_1 and f_2 are convex and the optimum solution range is $x \in [0, 2]$.

7.2 POL Function

In this function, $f_1(x) = \left[1 + (A_1 + B_1)^2 + (A_2 - B_2)^2\right]$, $f_2(x) = \left[(x_1 + 3)^2 + (x_2 + 1)^2\right]$, where $A_1 = 0.5\sin 1 - 2\cos 1 + \sin 2 - 1.5\cos 2$, $A_2 = 1.5\sin 1 - \cos 1 + 2\sin 2 - 0.5\cos 2$, $B_1 = 0.5\sin x_1 - 2\cos x_1 + \sin x_2 - 1.5\cos x_2$, $B_2 = 1.5\sin x_1 - \cos x_1 + 2\sin x_2 - 0.5\cos x_2$, the number of variables is equal to 2; the intervals for finding solutions are $x_1, x_2 \in [-\pi, \pi]$; this function is non-convex, and the optimum solution range is $x_1, x_2 \in [-3, 3]$.

7.3 FON Function

In this function:

$$f_1(x) = 1 - \exp\left(-\sum_{i=0}^{3}\left(x_1 - \frac{1}{\sqrt{3}}\right)^2\right), \quad f_2(x) = 1 - \exp\left(-\sum_{i=0}^{3}\left(x_1 + \frac{1}{\sqrt{3}}\right)^2\right),$$

the number of variables is equal to 3; the intervals for finding solutions are $x_1, x_2, x_3 \in [-4, 4]$; this function is non-convex and the optimum solution range is $x_1 = x_2 = x_3 \in \left[-1/\sqrt{3}, 1/\sqrt{3}\right]$.

The derivatives of these functions were obtained by using MATLAB, the expressions for them are mentioned in the Master's thesis [18].

8 Experimental Results

In this section, two different algorithms (the classical GA and the new DSC) are applied to minimize scalarization functions for the three functions (SCH, POL, and FON). Then, the results obtained for both algorithms are compared.

8.1 Comments on Applying the Genetic Algorithm

The GA Toolbox in MATLAB can be used to build a set of versatile routines for implementing a wide range of GA methods. In this section, the major procedures of the GA Toolbox are outlined [19].

For each of the three examples: SCH, FON and POL, we have applied the GA Toolbox algorithm with the following options [20]:

1. Population type specifies the type of the input to the fitness function. We used double vector.
2. Population size = 80 chromosomes.
3. Creation function = feasible population. "GA creates a random initial population using a creation function. We can specify the range of the vectors in the initial population in the Initial range field in Population options. Feasible population creates a random initial population that satisfies all bounds and linear constraints".
4. Initial population = default. "The algorithm begins by creating a random initial population, the default value of the Initial range in the Population options is the interval [0,1]".
5. Fitness scaling = Rank.
6. Selection = Roulette.
7. Mutation function = Uniform.— Uniform mutation is a two-step process. First, the algorithm selects a fraction of the vector entries of an individual for mutation, where each entry has a probability Rate of being mutated. "The default value of Rate is 0.01. In the second step, the algorithm replaces each selected entry by a random number selected uniformly from the range for that entry".
8. Crossover function = two point.
9. Stopping criteria = 1000 iterations.
10. Fitness limit = 0.01 is the threshold of stopping criteria.

8.2 SCH Function Using the GA and DSC Algorithms

Table 1 shows the results of applying the GA and DSC algorithms 100 times on the scalarization function for SCH. Here, and for the other examples, we only note the minimum, maximum and average values of s, the minimum and maximum values of all variables, and the minimum, maximum and average numbers of iterations. The results of all runs are presented in [18].

Table 1. The summary of 100 runs of the SCH function by using the GA and DSC algorithm

		$s(x_1)$	x_1	Iterations
GA	Min	0	0.0405164	1
	Max	0	1.93095859	4
	Average	0	–	2
DSC	Min	0	−0.00474	2
	Max	0.009477	2.000481	27
	Average	0.000155	–	8

For both algorithms, the success rate is equal to 100%, with the threshold for the value of s equal to 0.01.

8.3 POL Function Using the GA and DSC Algorithms

Table 2 shows the results of applying the GA and DSC algorithms 100 times on the scalarization function for POL.

Table 2. The summary of 100 runs of the POL function by using the GA and DSC algorithm

		$s(x_1,x_2)$	x_1	x_2	Iterations
GA	Min	0.00123	−2.85887	−2.0227	1
	Max	1.24557	2.57113	1.59023	1000
	Average	0.763332	–	–	924
DSC	Min	9.73E-06	−3.13117	−3.12568	2
	Max	0.009449	3.123923	2.55464	14
	Average	0.002552	–	–	3

For the GA, the success rate is equal to 8%, with the threshold for the value of s equal to 0.01. For the DSC, the success rate is equal to 100%, with the same threshold.

8.4 FON Function Using the GA and DSC Algorithm

Table 3 shows the results of applying the GA and DSC algorithms 100 times on the scalarization function for FON.

Table 3. The summary of 100 runs of the FON function by using the GA and DSC algorithm

		$s(x_1,x_2,x_3)$	x_1	x_2	x_3	Iterations
GA	Min	0.00041	0.01314	0.01314	0.01314	5
	Max	0.02645	0.57952	0.57709	0.58359	997
	Average	0.057842	–	–	–	486
DSC	Min	0.001047	−0.57547	−0.57926	−0.58121	11
	Max	0.054234	0.580235	0.575841	0.579259	985
	Average	0.008114	–	–	–	263

For the GA, the success rate is equal to 76%, with the threshold for the value of s equal to 0.01. For the DSC, the successful rate was equal to 94% with the same threshold.

It is worth mentioning that, for the GA, all the x-values were positive and no negative optimum values were shown. That is why the results did not achieve all values in the optimum range; just the positive ones.

8.5 Comparisons and Discussions

In this section, we make comparisons between the theoretical ranges of Pareto optimal solutions (taken from the literature) and the ranges of minimal points for scalarization functions for the three tested examples: SCH, POL and FON. It can be seen that the DSC algorithm is better than the GA especially in covering the theoretical optimal range by the numerically computed values. Table 4 shows the differences in the results obtained by applying the GA and DSC algorithms.

Table 4. The differences between the ranges by using the GA and the DSC algorithms on the SCH, POL and FON testing functions (100 runs)

	SCH function	POL function		FON function		
	x_1	x_1	x_2	x_1	x_2	x_3
GA	[0.0405164, 1.930958]	[−2.85887, 2.57113]	[−2.0227, 1.59023]	[0.01314, 0.57952]	[0.01314, 0.57709]	[0.01314, 0.58359]
DSC	[−0.00474, 2.000481]	[−3.13117, 3.123923]	[−3.12568, 2.55464]	[−0.57547, 0.580235]	[−0.57926, 0.575841]	[−0.58121, 0.579259]
Theoretical value	[0, 2]	[−3, 3]	[−3, 3]	$[-\frac{1}{\sqrt{3}}, 1/\sqrt{3}]$	$[-\frac{1}{\sqrt{3}}, 1/\sqrt{3}]$	$[-\frac{1}{\sqrt{3}}, 1/\sqrt{3}]$

In the first column in Table 4, we have a comparison between GA and DSC for SCH function, where the results show that the DSC algorithm has covered the optimal range, but the classical GA has not. The same is true for the first variable of the POL example, but for the second one, the DSC still has not covered the whole optimal range (but it is better than the GA). Also, for the FON function the optimum solutions found by the DSC include positive and negative numbers, but in the GA case, the results are only in the positive part of the optimal range.

Table 5 shows the differences between the average numbers of iterations when applying GA and DSC to the tested functions. It is clear that the DSC algorithm is much faster than the GA for the POL and FON examples.

Table 5. The differences between the average iterations numbers by using the GA and the DSC algorithms on the SCH, POL and FON testing functions

	Average iteration for SCH function	Average iteration for POL function	Average iteration for FON function
GA	2	924	486
DSC	8	3	263

Table 6 shows the differences between the success rates for the GA and DSC algorithms, it's clear that DSC is better than GA.

Table 6. The differences between the success rate by using the GA and the DSC algorithms on the SCH, POL and FON testing functions

	Success rate for SCH function	Success rate for POL function	Success rate for FON function
GA	100%	8%	76%
DSC	100%	100%	94%

9 Conclusions

The main conclusions of this paper are highlighted as the following:

- A new scalarization method for MOO problems has been described. It can effectively find the multi-objective solutions and after converting the problem to an SOO problem through some global scalarization function.
- We have applied the GA and DSC algorithms to find multiple solutions for the SOO problem obtained by scalarization. The results show that applying the DSC algorithm is more effective than applying the classical GA.
- The POL example shows the great advantage of the DSC algorithm in both average number of iterations and the success rate.

Recommendations for the future work:

- This research can be extended by using other options in the GA Toolbox instead of the ones we have used here.
- We can apply other test examples for minimizing the scalarization function by using GA and DSC algorithms.

Acknowledgments. The second author would like to thank the Ministry of Higher Education and Scientific Research (MOHESR), Iraq.

References

1. Coello, C.A.C.: Evolutionary Algorithms for Solving Multi-Objective Problems, 2nd edn. Springer, Science+Business Media, LLC All, Heidelberg (2007)
2. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. Wiley, New York (2001)
3. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Artificial Intelligence, 3rd edn. Springer, Heidelberg (1996)
4. Deb, K., Member, A., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
5. Rahmo, E., Studniarski, M.: A new global scalarization method for multiobjective optimization with an arbitrary ordering cone. Appl. Math. **8**, 154–163 (2017)
6. Al-Jawadi, R., Studniarski, M., Younus, A.: A New Genetic Algorithm Based on Dissimilarities and Similarities. Submitted for publication
7. Ehrgot, M., Gandibleux, X.: Multiple Criteria Optimization State of the Art Annoteted Bibligraphic Surveys. Kluwer Academic Publishers, Dordrecht (2003)
8. Ehrgott, M.: Multicriteria Optimization. Springer, Berlin (2005)
9. Ehrgott, M.: International Doctoral School Algorithmic Decision Theory: MCDA and MOO (2007). http://www.lamsade.dauphine.fr/~gold/COST_slides/HanLecture1_ME.pdf
10. Caramia, M., Dell'Olmo, P.: Multi-objective Optimization Managment in Freight Logistic Increasing Capacity (2008)
11. Voß, T., Beume, N., Igel, C.: Scalarization versus indicator-based selection in multi-objective CMA evolution strategies. In: IEEE Congress on Evolutionary Computation, pp. 3036–3043. IEEE (2008)
12. Man, K.F., Tang, K.S., Kwong, S.: Genetic algorithms: concepts and applications. IEEE Trans. Ind. Electron. **43**(5), 519–534 (1996)
13. Chudasama, C.: Comparison of parents selection methods of genetic algorithm for TSP. In: International Conference on Computer Communications and Networks, CSI- COMNET, pp. 85–87 (2011)
14. Iquebal, M.A.: Genetic Algorithms and their Applications: An Overview. Ph.D. Agricultural Stat. Roll No. 9068 I.A.S.R.I., Libr. Ave. New Delhi-110012, pp. 1–11
15. Liang, Y., Leung, K.: Genetic Algorithm with adaptive elitist-population strategies for multimodal function optimization. Appl. Soft Comput. J. **11**(2), 2017–2034 (2010)
16. Minami, M.: Weak Pareto-optimal necessary conditions in a nondifferentiable multiobjective program on a Banach space. J. Optim. Theory Appl. **41**(3), 451–461 (1983)
17. Clarke, F.H.: Optimization and Nonsmooth Analysis. Wiley, New York (1983)
18. Younus, A.: Converting a Multi-Objective Optimization Problem to a Single-Objective Optimization Problem by Using a New Scalarization Method. M.Sc. Thesis, University of Łódź (2016)
19. Optimization Toolbox, User's Guide. The MathWorks (2008)
20. How the Genetic Algorithm Works - MATLAB & Simulink, Optimization Toolbox, User's Guide. The MathWorks (2008). http://www.mathworks.com/help/gads/how-the-genetic-algorithm-works.html

Measuring Cognitive Workload in Arithmetic Tasks Based on Response Time and EEG Features

Małgorzata Plechawska-Wójcik, Magdalena Borys, Mikhail Tokovarov, and Monika Kaczorowska(✉)

Lublin University of Technology, Lublin, Poland
{m.plechawska,m.borys,m.tokovarov}@pollub.pl,
monika.kaczorowska@pollub.edu.pl

Abstract. The aim of the present paper is to verify whether the cognitive load can be evaluated through the analysis of the examined person's response time and extracted EEG signal features. The research was based on an experiment consisting of six intervals ensuring various cognitive load level of arithmetic tasks. The paper describes in details the analysis process including signal pre-processing with artifact correction, feature extraction and outlier detection. Statistical verification of EEG band differences, response time and error rate in intervals was realised. Statistical correlations were found between EEG features and response time, however, the correlation strength increased inside the groups of intervals of similar cognitive workload level. Evoked related potentials were also analysed and their results confirmed the statistical outcomes.

Keywords: EEG · Spectral analysis · Cognitive workload · FFT · ERP

1 Introduction

According to literature cognitive load may be referred to as the quantitative measure of the use of a limited amount of working memory [1]. Similarly to a computer, when the complexity of a performed task becomes too high the person has no more resources to deal with the other upcoming task. Estimating cognitive workload is a difficult yet potentially profitable task. The methods of evaluating the cognitive load may be divided into two major groups: subjective and objective [2]. The methods of the first group are based on non-direct measuring i.e. the participants have to fill-in questionnaires specifying what amount of mental effort the task required. The second group includes such techniques as EEG, eye-tracking and ECG. All these techniques are used to measure specific human physiological indexes, such as pupil diameter, duration of saccades and fixation (ET), ratio, amplitude and power of specific brain wave components (EEG) as well as parameters of the cardiogram (ECG).

Accurate measurement of cognitive load may be applied in many fields, and one of them is education. The application of cognitive load measurement would give the ability to monitor the state of concentration/deconcentration providing the quantitative measure of an educational programme's effectiveness. That might help to improve the educational programme as well as adapt it to a particular student. Other fields where cognitive load measurement may find application are enhancing BCI performance and better treatment of patients [1].

The present paper is focused on checking whether cognitive load can be evaluated through the analysis of EEG signal features and participant response time. This work is the continuation of our previous analysis including pupillometry response and its correlation with EEG features.

The paper is structured as follows: Sect. 2 contains a review of the existing literature covering the scope of cognitive load measurement, the next section describes the methods of EEG signal analysis, the fourth section is aimed at explaining the methodology of the experiment, while the fifth and sixth sections include the description of feature extraction as well as applied statistical analysis. The last section concludes the paper.

2 Related Work

Cognitive load has been measured and analysed by researchers for years. First attempts included mainly subjective measures such as subjective report and psychophysiological measurement [3, 4] as well as subjective scales [5]. Nowadays mental workload is usually assessed also with the use of objective measures based on biomedical signals. Among them one can find electroencephalogram (EEG), electrocardiogram, eye tracking, functional near infrared, or transcranial Doppler sonography [6] and pupillometry [7].

EEG is still among the most popular methods of cognitive load assessment. Different features originating from EEG records are based on spatial analysis with the distinction between different waveforms: alpha and theta [2, 8] as well as beta or delta [9]. Other EEG-based features are spatial filtering methods such as common spatial patterns, source power co-modulation [10, 11] or wavelet-complexity based measures associated with entropy [12]. Also Event-Related Potentials were analysed [13].

Research covers different types of studies and tasks. Among them there are visual word recognition [14], visual searching [15], arithmetical tasks [16, 17], watching or listening to lectures/learning material [18] or imaginary tasks [19].

3 EEG Analysis

Electroencephalography is one of the methods of examining brain activity. It relies on non-invasive measurement of specific waveforms emitted by the human brain. This activity is recorded using an amplifier of special type called electroencephalograph.

EEG is applied in the diagnosis of many nervous system diseases e.g.: epilepsy, insomnia, headaches and migraines, brain tumours and injuries. Nowadays EEG analysis is also applied in the construction of brain-computer interfaces (BCI). EEG signals are differentiated and they depend on the age and state of the person examined. There are several EEG waves occurring in a typical EEG recording. The most typical among them are: delta waves (δ), theta waves (θ), alpha waves (α), beta waves (β), and gamma waves (γ). Alpha waves (α) have a frequency of 8 Hz to 13 Hz. They occur when the eyes of the person examined are closed. Beta waves (β) are the wave type with a frequency from 13 to 30 Hz. They can be observed when the person's attention is focused. Gamma activity is associated with higher cognitive processes.

Brain activity is often analysed via the EEG. One of the most popular way of EEG analysis is based on the spectra. The transition from the original time domain to the frequency domain is often performed on the basis of the Fourier transform. The Fast Fourier transform (FFT) algorithm computes the discrete Fourier transform (DFT). The FFT manages to reduce the complexity of computing the DFT. This method was applied in the present paper in order to obtain EEG features for the analysis.

In the paper the P300 paradigm was also applied. It is based on the potential event-related phenomenon. P300 appears as a reaction to a specific stimulus. It might be detected during the multiple presence of stimuli. Such a stimulus needs to be presented to a user repeatedly in a sequence. The P300 signal is a potential detected at a time between 300 and 600 ms after the expected stimulus. It depends on the individual user's characteristics. The P300 potential might be detected only after signal averaging. P300 potentials are an example of event-related potentials; they are elicited in the process of decision making.

4 Methodology, Experimental Setup and Data Acquisition

A detailed description of the experiment is presented in our previous work [20]. The analysis based on data from 9 participants of the same sex, age and handedness. Their mean age was 22 years. The participants were instructed to minimise body movement to reduce potential artefacts in their EEG signals. The only movement performed was clicking the left button of the mouse.

The experimental task took the form of six intervals of equal duration composed of a series of arithmetic tasks. Each of the six intervals contained 17 arithmetic tasks at different level of difficulty. This difficulty level rose at each consecutive interval. Intervals were separated with breaks. Each single arithmetical task was composed of two images. The first one presented the task to solve, the second one was the answer. Each task was displayed totally for 5500 ms, including 1300 ms of delay before it. An arithmetic problem was displayed for 400 ms. The answer was displayed for 200 ms after 700 ms delay before it. The rest of the time was for the participant's reaction. Participants were asked to click the left mouse button if the answer was correct and do nothing in the other case. In a single interval there were 6–7 correct answers.

5 Data Analysis

5.1 Pre-processing

Mitsar EEG 201, the 21 channel EEG amplifier was applied to record the EEG data. The data was transmitted to the computer with the native software (EEG Studio) using the EEG amplifier which enables to record data and monitor the signal in real time.

EEG Studio was also used to run the experiment, synchronize the EEG data with the displayed stimulus and save the responses. The signal was gathered from 19 electrodes placed in a special EEG cap according to the 10–20 EEG standard. Two reference electrodes (A1, A2) were placed on the ears of the participant. In addition, the ground electrode was placed in the centre of the frontal lobe. Average reference montage was applied. The values off all electrodes (without reference electrode) were summed up and averaged. The difference between average and referential electrode was measured. The samples were recorded with the frequency of 500 Hz and processed in real time.

Pre-processing of the EEG signal was performed in the WinEEG software. The first step covered electrical cable noise reduction using basic band pass filter (0.1 Hz to 50 Hz). The next step covered further signal filtration with standard high-pass filter to remove noised signal below 3 Hz.

Next step covered artefact correction to filter signals originating from users movements as well as from equipment itself. Principal Component Analysis (PCA) was applied to remove artefacts. Depending on the basic level of the noise of each recording, two or three PCA components were chosen to be removed. The last step of preprocessing power spectrum of all the channels were generated with a full scale of EEG frequency band i.e., 3 Hz to 50 Hz to visualize the change in the frequency band as well as to select data to further analysis.

5.2 Feature Extraction

The first of the features to be extracted was the time of reaction, i.e. the length of the time interval between demonstrating the stimulus and reaction (left mouse click by a participant). This time was measured only if the participant answered the question correctly, i.e. if he/she clicked the button after the demonstration of a correct solution.

Amplitudes of specific components of brain waves are one of the main features for analysing an EEG signal [9].

Typically the frequency intervals to be analysed correspond to the bands of theta, alpha, beta-1 (low beta) and beta-2 (regular beta) waves. As the literature says, the theta component includes the waves with the frequencies from 4 to 7 Hz, while the alpha band covers the frequency interval between 8 and 13 Hz, the beta1 component corresponds to the frequencies from the interval of 13 to 22 and the frequencies of the waves constituting the beta2 component can be found in the interval from 23 to 30 Hz [9]. In the present research the frequency analysis was performed in Matlab by means of a special toolbox called EEGLab. Along with the amplitudes the dominant frequencies, i.e. the frequencies with the maximum amplitude, were used as a feature. Spectral analysis was performed on the signals, obtained as a result of averaging all the stimuli within one interval of the experiment.

Event related potentials may be used as another indicator and index of measuring cognitive load [13]. The event related potentials usually have lower amplitudes than the signals originating from artefacts. Along with that an EEG signal is often noised, so in order to obtain clearly visible ERP the averaging algorithm should be used [21]. For applying this algorithm a precisely planned experiment is required, i.e. the intervals between presenting the stimuli to the participant should be of the same length and long enough. That approach ensures appropriate time for both the participant to react and the signal to be registered correctly. In addition to planning, initial artefact correction should be performed. Samples having the values greater than six standard deviations should be replaced by the values of the mean. The idea of the averaging algorithm is presented in the formula:

$$x_i = \frac{1}{n}\sum\nolimits_{j=0}^{n} x_{ij}, \qquad (1)$$

where:

n	- the number of the samples per interval;
i	- the sample number in an interval;
j	- the number of the interval;
x_i	- the value of the i-th averaged sample;
x_{ij}	- the value of the i-th sample in the j-th interval.

In accordance with the law of large numbers, the influence of noise and accidental artefacts that had an amplitude lower than 6 standard deviations and hence were not rejected is eliminated to a great degree.

In the present paper the algorithm was implemented in Qt IDE with the use of the C++ programming language. Extensive open source libraries make it possible to process data effectively and write them to an excel file, which makes further processing and storage easier. Figure 1 presents an example of data before and after the use of the averaging algorithm. As can be seen, the level of noise is so high before averaging that it would be difficult for an observer to tell an evoked related potential (ERP) from noisy peaks. And on the contrary: after averaging, the ERPs can be distinguished quite easily.

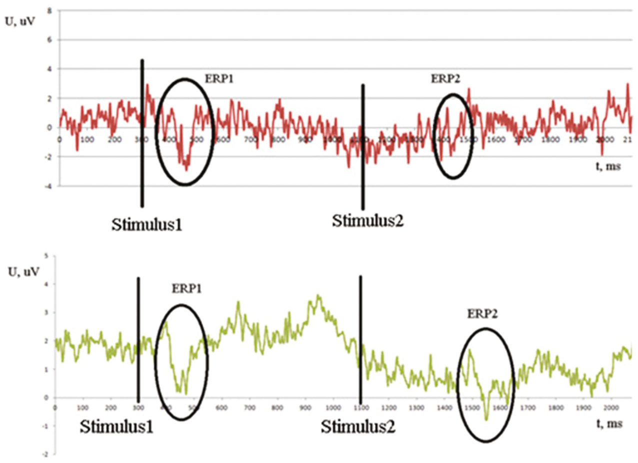


Fig. 1. Example of signal averaging. Top – row signal, bottom – averaged signal

6 Results

6.1 Statistical Analysis

Additional data such as the number of errors and a participant's mean response time to a stimulus in an interval were included in all extracted features from EEG data in each interval. Each set of features (76 EEG features plus 2 additional performance features) in an interval was treated as a single observation, thus there were 54 observations collected (9 participants $\times$ 6 intervals). However, in the last interval two participants did not react to the stimulus within required timeframe, therefore those two observations were excluded from analysis as not containing the required response to the stimulus. The other data (52 observations) were analysed statistically using the Statistica 13 (Dell Inc.) software. All tests were performed with the statistical significance level of 5%.

First of all, the relationship between features (EEG features as well as mean response time and number of errors) and subsequent intervals (as an ordinal variable since subsequent intervals were characterised by increasing the tasks' difficulty and thus increasing cognitive load) were investigated using Spearman's rank correlation coefficient. The correlation analysis confirmed the strong monotonic relationship (rho = 0.66, p-value = 0,0000001) between the number of errors committed by participants and the number of interval (as an ordinal variable) as well as the moderate relationship (rho = 0.3.44, p-value = 0,013) between the response time and the number of interval. The varying values of the number of errors and response time in intervals were presented graphically using box-and-whisker diagrams in Fig. 2.

Since there are relationships between the number of errors or response time and subsequent intervals, which indicate the increasing difficulty of tasks in intervals, the further analysis investigated the relationships of the number of errors and response time with EEG features, which reflects typical human brain activity.

Some of EEG features as well as the number of errors are not from a normally distributed population, which were tested using the Shapiro-Wilk test of normality. Therefore, for those features not only the parametric statistics model was used, but also an additional non-parametric statistical model was applied despite the central limit theorem to ensure validity of the results.

The increasing difficulty of tasks in intervals reflects the mean response time for the stimuli, and can therefore be treated as the indicator of participants' cognitive workload. Analysis of the relationship between mean response time and EEG features revealed 12 moderate and 10 weak linear (or at least monotonic as ensured by Spearman rho) correlations, presented in Table 1. The strongest correlations were found between response time and low beta wave measured in frontal-lobe channels (F4, Fz, F3). The correlations response time and low beta wave in F4 can be observed in Fig. 2. Nevertheless, the weaker correlations in frontal and central lobes with alpha (F4, F3, FP2, C4) and theta (FP2, F8, F4, Fz) waves can also be observed. Those results might be explained by two factors: increasing fatigue and excessive cognitive workload in consecutive intervals for participants, and low difficulty level of tasks in the initial intervals.

As can be observed in Fig. 2, the dispersions as well as values of mean response time in the first 3 and the last 3 intervals are alike. The analysis of variance (ANOVA) showed that in triples of task intervals (1–2–3 and 4–5–6) there are no significant differences in the mean values of the mean response time variable (mean response time did follow the normal distribution). The pairs of intervals 1–5, 1–6, 2–6, 3–4, 3–5, 3–6 were found significantly different using Fisher's Least Significant Differences method.

Therefore, the 27 observation from 1–2–3 intervals were marked as a low cognitive workload observation (Level 1) and the 25 observation from 4–5–6 intervals were marked as a high cognitive workload observation (Level 2).

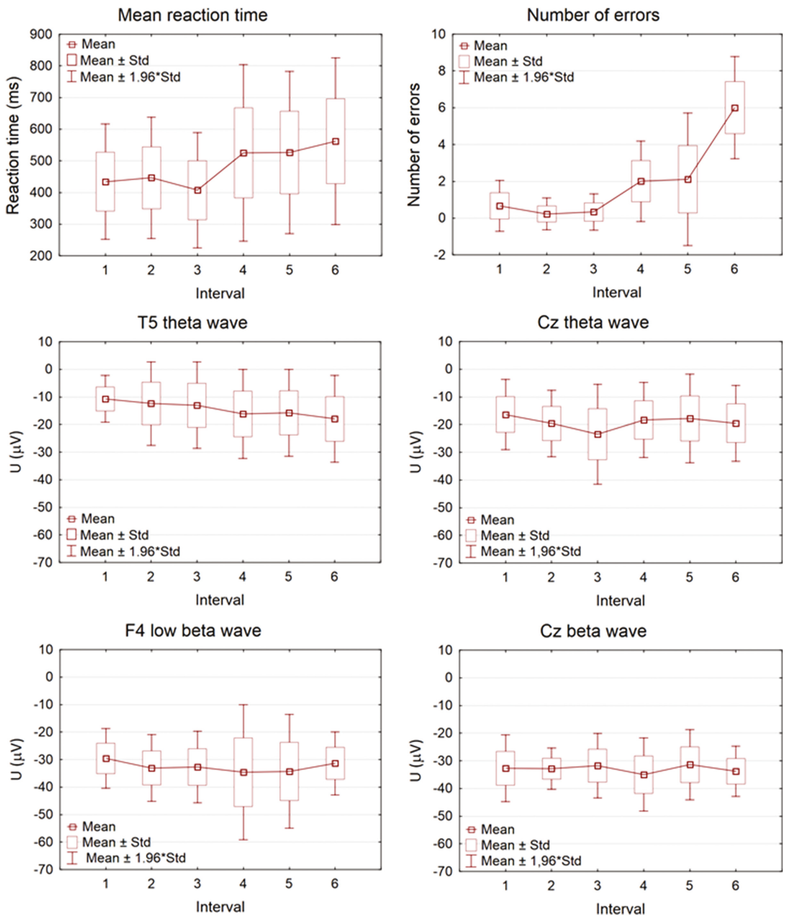


Fig. 2. The boxplots for response time, number of errors and selected EEG features

What is more, mean response times prove differences between two groups of intervals (Level 1 and Level 2). Response times at Level 1 occurred significantly lower than at Level 2. Even more informative are the number of errors results, which were clearly lower at Level 1. A large number of errors in Interval 6 gives a suspicion that participants found these tasks hard or even impossible to solve in the given time.

Analysis of the relationship between the mean response time and EEG features in low and high cognitive workload level groups show much stronger correlations than without groups (Table 2). In Level 1 group there are four strong correlations with low beta in F3, Fz, O2 and beta in Fz, three moderate correlations with low beta in F4, P4 and theta wave on Fz. In Level 2 group there are 24 strong or moderate correlations. Among those strong correlations, there are correlations with amplitudes of low beta wave in F4, Fz, beta wave in Cz, but also alpha wave in F3 and F4. Other correlations concerned low beta, beta and alpha waves, however there is no correlation with amplitudes of theta waves.

Table 1. The Pearson correlation coefficient value for mean response time with EEG features (with additional Spearman's rho for selected features without normal distribution)

Feature	Correlation coefficient r (or rho)	p-value
F4 low beta wave	0.444 (rho = 0.453, p-value = 0.001)	0.0009
Fz low beta wave	0.407	0.003
F4 alpha wave	0.386	0.005
F3 low beta wave	0.379	0.006
F8 theta wave	0.372 (rho = 0.352)	0.007
Cz beta wave	0.362 (rho = 0.343)	0.008
Fp2 theta wave	0.357	0.009
T4 beta waves	0.357	0.009
F3 alpha wave	0.325	0.019
Fz beta wave	0.312 (rho = 0.392, p-value = 0.01)	0.025
F4 theta wave	0.310 (rho = 0.283)	0.025
Fp2 low beta wave	0.306	0.027
F3 beta wave	0.296	0.033
Fp2 beta wave	0.294	0.034
Fp2 alpha wave	0.294	0.034
Fz theta wave	0.293 (rho = 0.340)	0.035
C4 low beta wave	0.293 (rho = 0.332)	0.035
C4 alpha wave	0.29	0.037
Fp1 theta wave	0.287	0.039
Cz low beta wave	0.282	0.043
F8 low beta wave	0.281	0.044

The independent two-sample t-test as well as the Mann-Whitney U test indicated that only the amplitude of the theta wave for the T5 channel in the Level 1 and Level 2 group has statistically different means (t = 2.25 with p-value = 0.029, U = 201 with p-value = 0.0127).

Table 2. The Pearson correlation coefficient value for mean response time with EEG features in two groups (with additional Spearman's rho for selected features without normal distribution)

Feature	Level 1		Level 2	
	Corr. coeff. r (or rho)	p-value	Corr. coeff. r (or rho)	p-value
F7 beta wave			0,486 (rho not significant)	0,014
F7 alpha wave			0,467	0,019
T3 beta wave			0,462	0,020
F3 low beta wave	**0,548**	0,003	0,429	0,032
F3 beta wave			0,487 (rho = 0.459)	0,014
F3 alpha wave	0,149	0,047	**0,567**	0,003
C3 alpha wave			0,454	0,023
Fz low beta wave	**0,569**	0,002	**0,548** (rho = 0.506)	0,005
Fz beta wave	**0,540**	0,004	0,432 (rho = 0.42)	0,031
Fz alpha wave			0,444	0,026
Cz low beta wave			0,423	0,035
Cz beta wave			**0,546**	0,005
Cz alpha wave			0,415	0,039
Fp2 low beta wave			0,402	0,047
Fp2 beta wave			0,452	0,023
Fp2 theta wave	0,399	0,039		
Fp2 alpha wave			0,409	0,042
F4 low beta wave	0,404	0,037	**0,622** (rho = 0.569)	0,001
F4 beta wave			0,403 (rho = 0.427)	0,046
F4 alpha wave			**0,502**	0,010
C4 low beta wave			0,408 (rho = 0.447)	0,043
C4 beta wave			0,471	0,017
C4 alpha wave			0,492	0,012
P4 low beta wave	0,478 (rho = 0.48)	0,012		
P4 beta wave			0,447	0,025
O2 low beta wave	**0,560** (rho = 0.472)	0,002		
T4 beta wave			**0,605**	0,001

The increasing difficulty of tasks in intervals also results in an increasing number of errors (Fig. 2). Analysis of the relationship between the number of errors and EEG features indicated only two significant monotonic correlations (Table 3) – negative correlation with the amplitude of the low beta wave from channel P3 and positive correlation with the amplitude of the theta wave from channel Fz. Correlation between increasing number of errors and theta waves most likely indicates tiredness or exhaustion of participants. Which is also confirmed by negative correlation with beta wave, indicating decrease of busy or anxious thinking and active concentration.

Table 3. Spearman's rho correlation coefficient value for number of errors with EEG features

Feature	Correlation coefficient rho	p-value
P3 low beta waves	–0,349	0.012
Fz theta waves	0,308	0.026

Additionally, Kendall's tau coefficients were also calculated indicating weak correlation (0,18 < tau < 0.26) between the number of errors and low beta wave in P3 and T5 (negative), beta wave in T5 (negative), theta wave in Fz, C3 and Cz as well as theta wave in T5 (negative, tau = –0.19). Results for T5 electrode are consistent with the previously discussed results of independent two-sample t-test and the Mann-Whitney U test for groups based on the level of cognitive workload (Fig. 2).

Using the number of errors, three group were created: Group 1 with no errors (20 observations), Group 2 with 1 or 2 errors (20 observations) and Group 3 with more than 2 errors (12 observations). The Kruskal-Wallis one-way analysis of variance showed that in groups there are significant differences in the median values of the number of errors variable (number of errors did not follow the normal distribution). The pairs of groups 1–2, 1–3, 2–3 were found significantly different using Fisher's Least Significant Differences method.

The Kruskal-Wallis one-way analysis of variance indicated the following significant differences within the group, which were further investigated using Fisher's Least Significant Differences method as a post-hoc test:

– amplitude of beta wave in Cz H(2, N = 52) = 10,31 with p-value = 0,006, significant difference between Group 1–2 and 2–3 (see Fig. 2);
– amplitude of theta wave in Cz H(2, N = 52) = 10,22 with p-value = 0,006, significant difference between Group 1–2 (see Fig. 2);
– amplitude of low beta wave in T6 H(2, N = 52) = 8,1 with p-value = 0,0174, significant difference between Group 1–2;
– amplitude of low beta wave in T5 H(2, N = 52) = 7,6 with p-value = 0,022, significant difference between Group 1–2;
– amplitude of low beta wave in P3_beta1 H(2, N = 52) = 7,37 with p-value = 0,025, significant difference between Group 1–3.

These results prove a significant influence of beta waves in the process of cognitive load and the change of its intensification. Error number confirmed to be a significant indicator of cognitive workload applied in the arithmetical tasks. What is more, the number of errors gives additional insights into cognitive processing.

6.2 Evoked Related Potentials

Evoked Related Potentials (ERP), also known as time-locked EEG activity [22], is defined as any stereotyped electrophysiological response to a stimulus (internal or external). In the analysis, external visual stimuli were presented. Single stimulus was complex, composed of two figures presented to participants at specified intervals. The first one covered a mathematical task, whereas the second one contained the response. The first figure was presented with 300 ms delay and its exposition time was 400 ms.

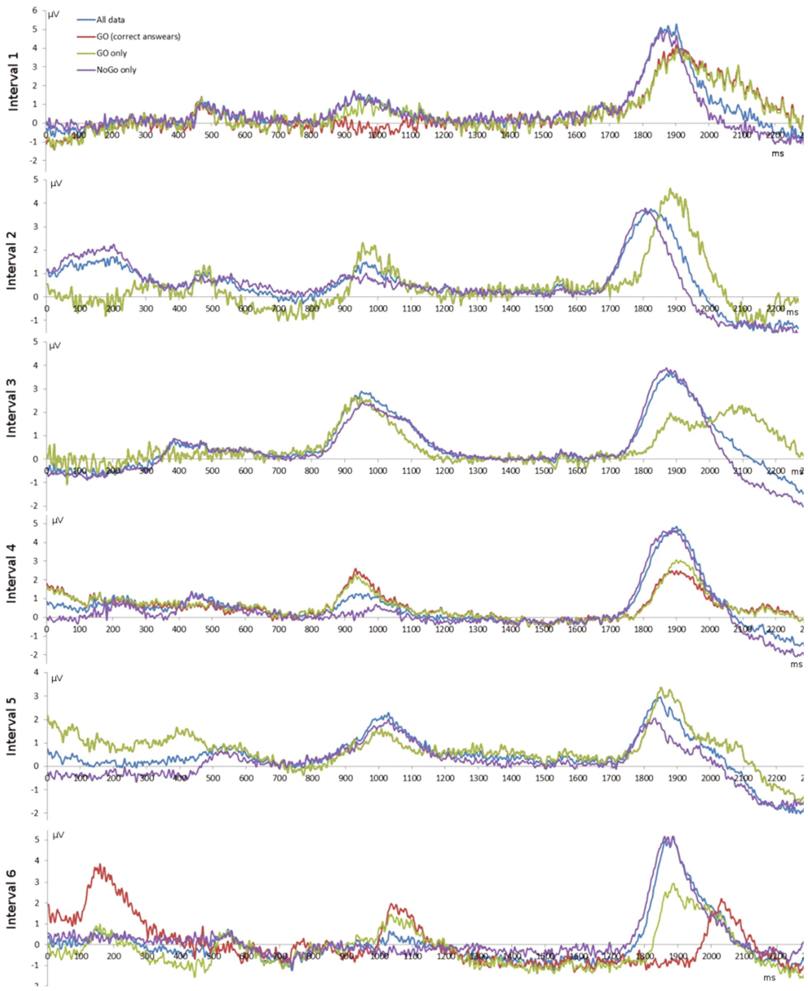


Fig. 3. ERP results for participant no. 9, electrode F7

The second figure was presented 1100 ms after the first one, with the exposition time of 200 ms.

Figure 3 presents signals in the time domain averaged after all the stimuli in a single interval for a single participant. Figures present only the most important part of the signal containing the stimulus presentation and the participant response. The rest of the signal, representing a break between stimuli, was skipped.

Figure 3 presents four signals differing in the type of response: GO task (where the displayed response was true – green line), GO task with correct participant response (red line), NoGO task (where the displayed response was false – navy blue line), and all data (blue line).

The example data for one participant were selected as representative of several channels (F8, FP1, FP2, F3, F4, C3, Fz, Cz) for which similar results were obtained.

Results clearly show moments of participants' reaction to presented stimuli. The first peak presents participant involuntary reaction on a displayed figure (arithmetic task). There are no noticeable differences between all four signals. The second peak represents participants' reaction on the second figure displaying the result. The response to all signals occurs earlier, but the response to the correct answer is visible about 300 ms later, which coincides with the P300 paradigm and proves that the reaction is conscious. What is more, there are visible differences between intervals. The delay in the first interval is shorter than the delay in the second and third interval, which correlates with the increasing difficulty level. This discrepancy is clearly seen in the third interval. Weak reaction in the sixth interval might prove that the tasks were too hard for the participant who was exhausted with the task.

7 Discussion

The paper discussed the possibility of cognitive load evaluation through the analysis of the response time and EEG signal features. Several factors result from the presented analysis. Firstly, the strongest correlations were found between reaction time and low beta wave measured in frontal lobe channels. Nevertheless, weaker correlations with alpha and theta waves were also observed, which can be explained by increasing fatigue and excessive cognitive load in consecutive intervals for some participants. Secondly, the analysis of relationship between the number of errors and EEG features indicated only two monotonic correlations – negative correlation with amplitude of low beta wave from channel P3 and positive correlation with amplitude of theta wave from channel Fz. Correlation between increasing number of errors and theta band indicated rising tiredness or exhaustion of participants. This is also confirmed by negative correlation with beta wave.

The analysis was also performed between two cognitive workload level groups. Study of the relationship between mean reaction time and EEG features showed much stronger correlations than without groups. In Level 1 group there are four strong correlations with low beta in F3, Fz, O2 and beta in Fz, and three moderate correlations with low beta in F4, P4 and theta wave in Fz. In Level 2 group there are 24 strong or moderate correlations: low beta wave in F4, Fz, beta wave in Cz, and alpha wave in F3 and F4. Moderate correlations include low beta, beta and alpha waves.

The Kruskal-Wallis one-way analysis of variance indicated the following significant difference within groups related to error numbers in 1 amplitude of beta, 1 amplitude of theta and for 3 amplitudes of low beta wave. These differences occur on different areas of the brain than the results obtained earlier in the study.

The results of evoked related potentials evidenced participant reaction for both figures included in a stimulus. Results prove fast (100–150 ms) involuntary reaction to the displayed figures and longer, conscious reaction to the displayed result. Reaction delay in initial intervals (for example Interval 1) was shorter due to almost instant, automatic answer. These delays were larger in further intervals (Interval 2–3).

Intervals 5 and 6 occurred to be hard for participants. Weak, inconclusive responses might be related to the fatigue and exhaustion of participants.

References

1. Gevins, A., Smith, M.E., McEvoy, L., Yu, D.: High-resolution EEG mapping of cortical activation related to working memory: effects of task difficulty, type of processing, and practice. Cereb. Cortex **7**, 374–385 (1997). doi:10.1093/cercor/7.4.374
2. Kruger, J., Doherty, S.: Measuring cognitive load in the presence of educational video: towards a multimodal methodology. Australas. J. Educ. Technol. **32**, 19–31 (2016). doi:10.14742/ajet.3084
3. Hancock, P.A., Chignell, M.H.: Mental workload dynamics in adaptive interface design. IEEE Trans. Syst. Man. Cybern. **18**, 647–658 (1988)
4. O'Donnell, R.D., Eggemeier, F.T.: Workload assessment methodology. In: Handbook of perception and human performance, vol. 2. Cognitive Processes and Performance. Wiley (1986)
5. Hart, S.G.: Nasa-Task Load Index (NASA-TLX); 20 years later. Proc. Hum. Factors Ergon. Soc. Annu. Meet **50**, 904–908 (2006). doi:10.1177/154193120605000909
6. Matthews, G., Reinerman-Jones, L.E., Barber, D.J., Abich IV, J.: The psychometrics of mental workload: multiple measures are sensitive but divergent. Hum. Factors **57**, 125–143 (2015)
7. Ren, P., Barreto, A., Huang, J., Gao, Y., Ortega, F.R., Adjouadi, M.: Off-line and on-line stress detection through processing of the pupil diameter signal. Ann. Biomed. Eng. **42**, 162–176 (2014). doi:10.1007/s10439-013-0880-9
8. Kumar, N., Kumar, J.: Measurement of cognitive load in HCI systems using EEG power spectrum: an experimental study. Proc. Comput. Sci. **84**, 70–78 (2016). doi:10.1016/j.procs.2016.04.068
9. Zarjam, P., Epps, J., Chen, F.: Spectral EEG features for evaluating cognitive load. Conf. Proc. IEEE Eng. Med. Biol. Soc. **2011**, 3841–3844 (2011). Annual Conference
10. Brouwer, A., Hogervorst, M.A., van Erp, J.B.F., Haufe, S., Kim, J., Kim, I., Acqualagna, L., Bosse, S., Porbadnigk, A.K., Marchal-crespo, L., Zimmermann, R., Lambercy, O., Schultze-kraft, M., Dähne, S., Gugler, M.: Unsupervised classification of operator workload from brain signals This. J. Neural Eng. **13**, 1–15 (2016). doi:10.1088/1741-2560/13/3/036008
11. Chang, H., Hung, I., Chew, S.W., Chen, N.: Yet another objective approach for measuring cognitive load using EEG-based workload, 501–502 (2016). doi:10.1109/ICALT.2016.145
12. Zarjam, P., Epps, J., Chen, F., Lovell, N.H.: Classification of working memory load using wavelet complexity features of EEG signals. In: Huang, T., Zeng, Z., Li, C., Leung, C.S. (eds.) Neural Information Processing: 19th International Conference, ICONIP 2012, Doha, Qatar, 12–15 November 2012, Proceedings, Part II, pp. 692–699. Springer, Heidelberg (2012)
13. Sereno, S.C., Rayner, K.: Measuring word recognition in reading: eye movements and event-related potentials. Trends Cogn. Sci. **7**, 489–493 (2003). doi:10.1016/j.tics.2003.09.010
14. Marshall, S.P.: The index of cognitive activity: measuring cognitive workload. In: Proceedings of IEEE 7th Conference on Human Factors and Power Plants, pp. 5–9 (2002). doi:10.1109/HFPP.2002.1042860

15. Porter, G., Troscianko, T., Gilchrist, I.D.: Effort during visual search and counting: insights from pupillometry Gillian. Q. J. Exp. Psychol. **60**, 211–229 (2007). doi:10.1080/17470210600673818
16. Ryu, K., Myung, R.: Evaluation of mental workload with a combined measure based on physiological indices during a dual task of tracking and mental arithmetic. Int. J. Ind. Ergon. **35**, 991–1009 (2005). doi:10.1016/j.ergon.2005.04.005
17. Zarjam, P., Epps, J., Lovell, N.H.: Beyond Subjective Self-Rating: EEG Signal Classification of Cognitive Workload (2015)
18. Kruger, J.-L., Hefer, E., Matthew, G.: Measuring the impact of subtitles on cognitive load : eye tracking and dynamic audiovisual texts. In: Proceedings of Eye Tracking South Africa, pp. 6–11 (2013)
19. Rozado, D., Duenser, A.: Combining EEG with pupillometry to improve cognitive workload detection. Comput. (Long. Beach. Calif). **48**, 18–25 (2016). doi:10.1109/MC.2015.314
20. Borys, M., Tokovarov, M., Wawrzyk, M., Wesołowska, K.: An Analysis of eye-tracking and electroencephalography data for cognitive load measurement during arithmetic tasks, pp. 287–292 (2017)
21. Haapalainen, E., Kim, S., Forlizzi, J.F., Dey, A.K.: Psycho-physiological measures for assessing cognitive load. In: Proceedings of the 12th ACM International Conference on Ubiquitous Computing, pp. 301–310. ACM, New York (2010)
22. Light, G.A., Williams, L.E., Minow, F., Sprock, J., Rissling, A., Sharp, R., Swerdlow, N.R., Braff, D.L.: Electroencephalography (EEG) and event-related potentials (ERPs) with human participants. Curr. Protoc. Neurosci. 6–25 (2010)

A Method for Genetic Selection of the Dynamic Signature Global Features' Subset

Marcin Zalasiński(✉) and Krzysztof Cpałka

Institute of Computational Intelligence,
Częstochowa University of Technology, Częstochowa, Poland
{marcin.zalasinski,krzysztof.cpalka}@iisi.pcz.pl

Abstract. Dynamic signature is a biometric attribute which can be used to perform verification of the identity. It consists of waveforms describing dynamics of a signing process. The waveforms are acquired using a digital input device, e.g. graphic tablet or touchscreen. During verification process the signature is usually represented by descriptors, which can be so-called global features. In this paper, we propose a new genetic approach to select a specified number of the most characteristic global features for each signer, which are used in the identity verification process. Proposed method was tested using known dynamic signatures database - MCYT-100.

Keywords: Biometric system · Dynamic signature verification · Genetic algorithm · Automatic features' selection

1 Introduction

Dynamic signature is a biometric attribute which bases on a characteristic, learned behavior of an individual [1, 6, 8]. It is described by waveforms which contain information about dynamics of a signing process (e.g. pen position, pen pressure, pen angle). Acquisition of the dynamic signature can be performed using digital input device, e.g. graphic tablet or any device equipped with a touch screen.

The dynamic signature is used for identity verification. At the stage of verification process it is usually represented by descriptors extracted from the waveforms. They can be e.g. so-called global features [5, 7, 10, 21], which are scalar values describing properties of the signing process, e.g. total time of the process, number of pen-ups, etc.

Our previous research has shown that classification error of the verification process depends on the number of used global features [22]. Due to this, in this paper we are focus on selection of a specified number of the most characteristic global features individually for each user. This process will be performed using a genetic algorithm with properly defined adaptation function. Efficiency of the selection process was tested using MCYT-100 dynamic signature database [13].

The structure of the paper is as follows: Sect. 2 presents data representation used by the algorithm, in Sect. 3 a description of the proposed method for genetic selection of the global features is presented, in Sect. 4 simulation results are presented and in Sect. 5 conclusions are drawn.

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 38th International Conference on Information Systems Architecture and Technology – ISAT 2017*, Advances in Intelligent Systems and Computing 655, DOI 10.1007/978-3-319-67220-5_7

2 Data Representation in the Algorithm

In this section data representation used by the algorithm is shown. The algorithm uses a set of N so-called global features, which has been defined in [5]. Each signer i creates J training signatures during training phase, so at this stage we have $J \cdot N$ global features for the signer i. They can be stored in the matrix $\mathbf{G}_i$, which has the following structure:

$$\mathbf{G}_i = \begin{bmatrix} g_{i,1,1} & \cdots & g_{i,N,1} \\ \vdots & & \vdots \\ g_{i,1,J} & \cdots & g_{i,N,J} \end{bmatrix}, \tag{1}$$

where $g_{i,n,j}$ is the value of the global feature $n\,(n = 1,\ldots,N)$ determined for the signature $j\,(j = 1,\ldots,J)$ created by the user $i\,(i = 1,\ldots,I)$.

To simplify the description, we can average the values of the corresponding global features describing reference signatures of the user i. Averaged values of the features can be stored in the vector $\bar{\mathbf{g}}_i = \left[\bar{g}_{i,1},\ldots,\bar{g}_{i,N}\right]$, where $\bar{g}_{i,n}$ is the average value of n-th global feature for all J reference signatures of the user i:

$$\bar{g}_{i,n} = \frac{1}{J}\sum_{j=1}^{J} g_{i,n,j}. \tag{2}$$

3 Genetic Selection of the Global Features

This section describes assumption of the genetic selection algorithm. For each signer, we select the most characteristic global features, which are used in the verification phase. The number of features should be close to the specified number *Nfeat*. For this purpose, we use genetic algorithm with properly defined adaptation function. Considered procedure works according to the algorithm shown in Fig. 1.

3.1 Encoding of Solutions

Each individual $\mathbf{X}_{i,ch}$ from the population encodes a full set of global features of the dynamic signature of the user i and it is expressed as follows:

$$\mathbf{X}_{i,ch} = \left\{\bar{g}_{i,1},\ldots,\bar{g}_{i,N}\right\} = \left\{X_{i,ch,1},\ldots,X_{i,ch,N}\right\}, \tag{3}$$

where each gene of the individual $\mathbf{X}_{i,ch}$ encodes information whether global feature related to this gene would be included to the subset of the most characteristic descriptors of the dynamic signature of the user i (value of each gene is equal to 0 or 1), ch is the number of chromosome $(ch = 1, 2,\ldots,Ch)$.

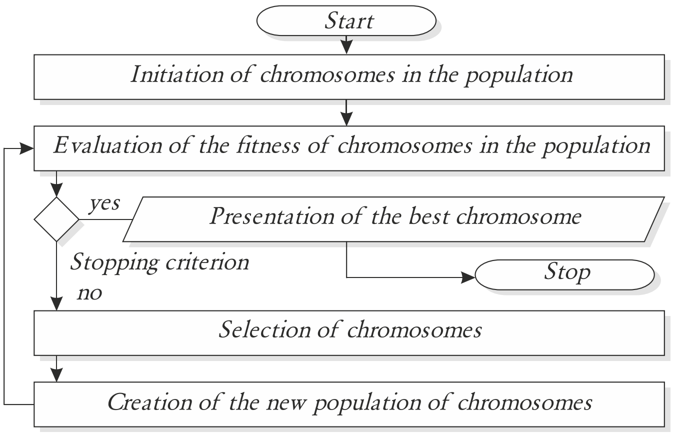

Start
Initiation of chromosomes in the population
Evaluation of the fitness of chromosomes in the population
yes
Presentation of the best chromosome
Stopping criterion
no
Stop
Selection of chromosomes
Creation of the new population of chromosomes

Fig. 1. Scheme of the genetic algorithm.

3.2 Processing of Solutions

A purpose of the genetic algorithm [4, 9, 16] is selection of about *Nfeat* the most characteristic global features which values determined for training signatures are similar. This process is performed for each user. First, random initialization of chromosomes is performed. Next, they are evaluated by a properly defined evaluation function (see Sect. 3.3. Next, stopping criterion is checked. In our algorithm, it depends on specified number of steps (generations) performed by the algorithm. If the stopping criterion is satisfied, then the procedure of evolutionary features selection quits and returns the information about the best chromosome from the population. If the criterion is not satisfied, evolution of the population is performed. At this stage, we are using roulette wheel method to select the best adapted individuals and classic genetic operators - crossover and mutation - to create new population [14] which replaces the old one. Next, the whole process is repeated.

3.3 Evaluation of Solutions

To determine value of fitness function we use L_i global features (descriptors) chosen during genetic selection process. They are descriptors for which values of genes in the chromosome $\mathbf{X}_{i,ch}$ are equal 1. To simplify description of determination of the fitness function values, the following variables will be used:

- $\mathbf{d}_{i,j} = \left[d_{i,j,1}, \ldots, d_{i,j,L_i}\right]$ - the vector containing the values of the selected L_i descriptors of the signature j of the user i,
- $\bar{\mathbf{d}}_i = \left[\bar{d}_{i,1}, \ldots, \bar{d}_{i,L_i}\right]$ - the vector containing averaged in the context of the reference signatures of the user i values of selected L_i descriptors,
- $\mathbf{D}_i = \left[\mathbf{d}_{i,1}, \ldots, \mathbf{d}_{i,J}\right]$ - the matrix containing values of selected L_i descriptors of J reference signatures of the user i.

Value of adaptation function is based on two weighted criteria: the similarity between training signatures and the number of selected features. Therefore, the purpose of the algorithm is selection of individuals $\mathbf{X}_{i,ch}$, which:

– Encode features which values determined for training signatures are similar. This approach results from the fact that in the discussed problem values of the evaluation function of individuals cannot be directly determined by, for example, the value of the classification error. It would be possible to determine the error, but then the complexity of the algorithm used for features selection would increase significantly and its scalability would decrease. Scalability of a biometric algorithm is dependent on, among others, how easily we can add new users to the database containing biometric characteristics and the number of the users in the database (determination of descriptors of new users' signatures should not include the signatures of existing users). For this reason, the definition of a function evaluating individual subsets of features is based on the assumption that features whose values are as similar as possible in particular training sessions (sessions during which the reference signatures are created) are higher evaluated. Apart from using the homogeneity of features' values in sessions, the heterogeneity between characteristic values of different users' features could be considered, but in biometric algorithms this approach should not be used (as it has been written previously).
– Encode the number of features close to the specified number. The aim of this approach is to increase the effectiveness of feature reduction and to select only these features which are the most characteristic for reference signatures of individual users.

The similarity is determined using Mahalanobis distance between values of selected global features of all reference signatures of the user i and their average values. It is denoted as $MD_i \in [0, +\infty]$ and defined as follows:

$$MD_i = \frac{1}{J}\sum_{j=1}^{J} \sqrt{\left(\mathbf{d}_{i,j} - \bar{\mathbf{d}}_i\right)\left(\operatorname{cov}(\mathbf{D}_i)\right)^{-1}\left(\mathbf{d}_{i,j} - \bar{\mathbf{d}}_i\right)^{\mathrm{T}}}. \tag{4}$$

Next, the distance should be compared to the specified maximum distance MDS, which is set as a constant. It is performed to normalize value of MD_i to the unit range. In considered problem normalization can be performed by use of membership function defined as follows (Fig. 2a):

$$\mu(MD_i) = \begin{cases} \frac{MDS - MD_i}{MDS - 0} & \text{for} \quad 0 \le MD_i \le MDS \\ 0 & \text{for} \quad MD_i > MDS. \end{cases} \tag{5}$$

The number of selected features $L_i \in [0, N]$ should be compared to the specified number of features $Nfeat$, which is set as a constant. It is performed to normalize value of L_i to the unit range. In considered problem normalization can be performed by use of membership function defined as follows (Fig. 2b):

$$\mu(L_i) = \begin{cases} \frac{L_i}{Nfeat} & \text{for} \quad 0 \le L_i \le Nfeat \\ \frac{N-L_i}{N-Nfeat} & \text{for} \quad Nfeat < L_i \le N. \end{cases} \tag{6}$$

The purpose of the algorithm is maximization of adaptation function $\mathrm{ff}(\mathbf{X}_{i,ch})$. The value of the function is calculated using weighted algebraic triangular norm $T^*\{\cdot\}$ [2, 14]. It can be used due to the normalization of components MD_i and L_i. It is defined as follows:

$$\begin{aligned} \mathrm{ff}\left(\mathbf{X}_{i,ch}\right) &= T^*\{\mu(MD_i), \mu(L_i); w_{MD}, w_L\} \\ &= T\{1 - w_{MD} \cdot (1 - \mu(MD_i)), 1 - w_L \cdot (1 - \mu(L_i))\} \\ &= (1 - w_{MD} \cdot (1 - \mu(MD_i))) \cdot (1 - w_L \cdot (1 - \mu(L_i))), \end{aligned} \tag{7}$$

where t-norm $T\{\cdot\}$ is a generalization of the usual two-valued logical conjunction (studied in classical logic), $w_L \in [0, 1]$ and $w_{MD} \in [0, 1]$ mean weights of importance of the arguments $\mu(MD_i)$ and $\mu(L_i)$.

After selection of the most characteristic global features, signature verification is performed.

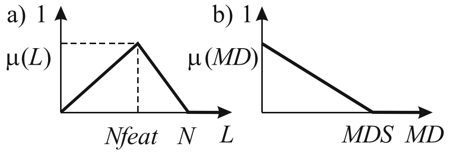


Fig. 2. Membership function used to evaluate selected features based on: (a) the similarity criterion, and (b) the selected features number criterion.

3.4 Signature Verification

Process of signature verification is performed using flexible fuzzy system presented in [18–20], which is one-class classifier. The system has $2 \cdot L$ inputs, which contain information about similarities between test signature and training signatures' global features, and weights of importance calculated for each global feature. The weights are calculated using similarity of training signatures' features and their dispersion. The system has also one output, which defines genuineness of the test signature. If this value is higher than a threshold value, the signature is considered to be true. General schema of the system is presented in Fig. 3, its detailed description can be found in our previous papers, e.g. [3, 18, 19].

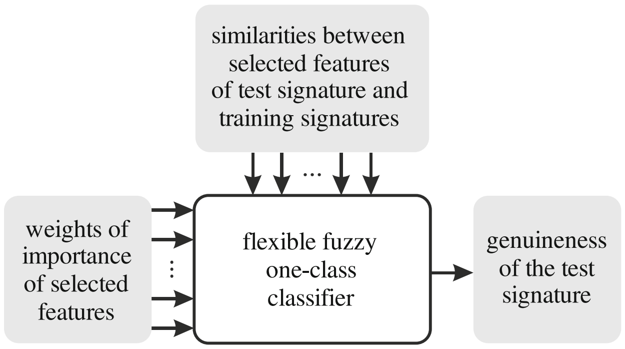

similarities between selected features of test signature and training signatures
weights of importance of selected features
flexible fuzzy one-class classifier
genuineness of the test signature

Fig. 3. A general schema of the flexible fuzzy one-class classifier.

4 Simulation Results

During simulations, we selected the specified number of the most characteristic global features for each user available in considered dynamic signature database. We used public MCYT-100 dynamic signature database [13]. Next, we performed identity verification process using selected global features to compare efficiency of our method to other approaches for the dynamic signature verification using global features. During the simulations, the following assumptions were adopted:

– In the training phase, we used 5 randomly selected genuine signatures of each signer.
– In the test phase, we used 15 genuine signatures and 15 forged signatures (so-called skilled forgeries) of each signer.
– Population contained 100 chromosomes.
– Crossover was performed with probability equal to 0.8 at three points.
– Mutation was performed for each gene with probability equal to 0.02.
– Algorithm stopped after the lapse of a determined number of 250 generations.
– Number of specified features *Nfeat* was equal 50 for each user, because our previous research prove that this value was optimal for the users from MCYT-100 database (see [22]).
– Weight w_{MD} value was set to 0.7.
– Weight w_L value was set to 0.3.

Conclusions from simulations can be summarized as follows:

– Efficiency of identity verification obtained by our method is presented in Table 1. The efficiency is measured using Equal Error Rate coefficient which specifies percentage number of signatures which were classified incorrectly [17]. The table also contains the results achieved by other methods for the dynamic signature verification which took into account global features, used maximum 5 reference signatures and were tested using so-called skilled forgeries. You can see that the proposed method works with a good accuracy.

Table 1. Comparison of the results for the dynamic signature verification methods based on global features which use maximum 5 reference signatures and were tested using so-called skilled forgeries.

Method	Average EER
Nanni, Lumini (b) [11]	8.40%
Nanni, Lumini (a) [12]	7.60%
Fierrez-Aguilar et al. [5]	5.61%
Nanni [10]	5.20%
Lumini, Nanni [7]	4.50%
Zalasiński et al. [22]	2.92%
Our method	2.20%

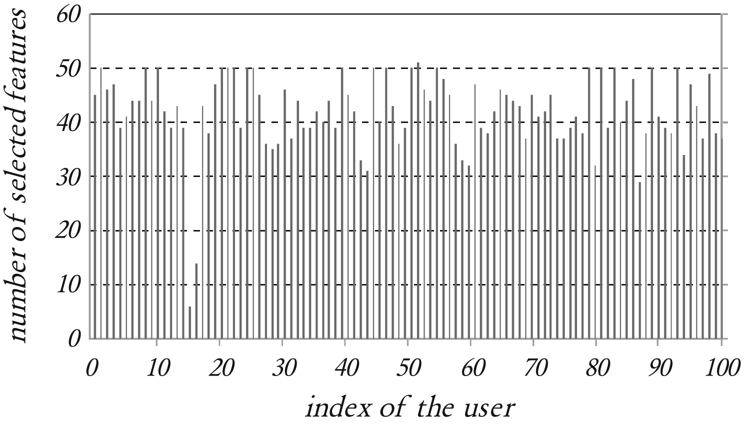


Fig. 4. The number of selected features of the dynamic signature for users of the MCYT-100 database.

– We also present a number of selected global features for each user (see Fig. 4). The figure shows that the algorithm usually selected features number lower than specified optimal value (50 from 100 features), but not less than 50%. It shows that the algorithm seeks to reach a compromise between selection of specified number of features and features characterized by the best similarity in the context of training signatures.
– The algorithm selected different subsets of features for individual users. Itmeans that in the considered subset of features there are not the ones which arethe best in the context of all users from used MCYT-100 database. Moreover, exemplary value of the evaluation function used for selection of features is presented in Fig. 5.
– The algorithm was also executed without selection of features. It means that for individual users all available features were considered. In that case accuracy of the algorithm was lower. Detailed discussion has been presented in paper [22]. The disadvantage of this approach was that values of some features were similar for

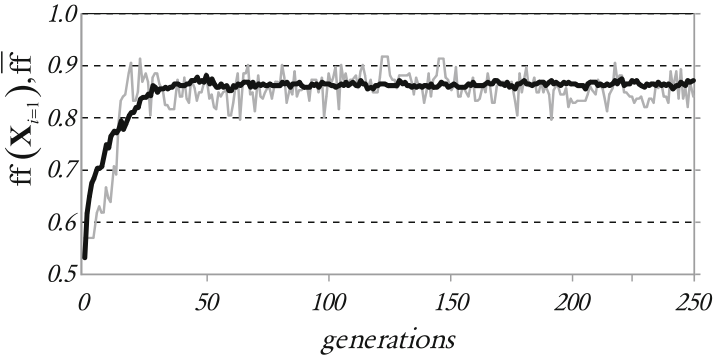


Fig. 5. Evaluation function of the form (7) for the best individual in the population during evolution process for exemplary user denoted by index 1 (gray line) and average value of the function for all users from MCYT-100 database (black line).

all considered users and classification accuracy was lower. The algorithm proposed in this paper solves these drawbacks.

– The algorithm was also executed without minimization of features' number. It means that the component $\mu(L_i)$ was omitted in function (7) (its weight w_L was equal 0). In that case accuracy of the algorithm was lower. Detailed discussion has been presented in paper [15]. The algorithm selected a subset of features for individual but the disadvantage of this approach was that the reduction level of features was not satisfactory. The algorithm proposed in this paper for some users (e.g. user 15, Fig. 4) selects only few the most characteristic features.

5 Conclusions

In this paper, a new method for genetic selection of the specified number of the most characteristic global features describing the dynamic signature was presented. The method used two criteria based on similarity of training signatures and specified number of selected features. Their importance was denoted by weights of importance. Simulation results show that selection of the specified number of the most characteristic features has a positive effect on the identity verification process.

Moreover, presented method can be used to resolve many other classification issues which allow us to select the optimal number of the most characteristic features describing considered pattern.

During our further research on the global features of the dynamic signature we are planning to develop, among others, different methods for evaluation of individual features and selection of them.

Acknowledgment. The project was financed by the National Science Centre (Poland) on the basis of the decision number DEC-2012/05/B/ST7/02138. The work presented in this paper was also supported by the grant number BS/MN 1-109-301/16/P.

The authors would like to thank the reviewers for very helpful suggestions and comments in the revision process.

References

1. Abhishek, S., Suresh, S.: An enhanced contextual DTW based system for online signature verification using Vector Quantization. Pattern Recogn. Lett. **84**, 22–28 (2016)
2. Cpałka, K.: Design of Interpretable Fuzzy Systems. Springer, Cham (2017)
3. Cpałka, K., Zalasiński, M., Rutkowski, L.: A new algorithm for identity verification based on the analysis of a handwritten dynamic signature. Appl. Soft Comput. **43**, 47–56 (2016)
4. El-Samak, A.F., Ashour, W.: Optimization of traveling salesman problem using affinity propagation clustering and genetic algorithm. J. Artif. Intell. Soft Comput. Res. **5**, 239–245 (2015)
5. Fierrez-Aguilar, J., Nanni, L., Lopez-Penalba, J., Ortega-Garcia, J., Maltoni, D.: An on-line signature verification system based on fusion of local and global information. In: Audio-and Video-based Biometric Person Authentication. Lecture Notes in Computer Science, vol. 3546, pp. 523–532 (2005)
6. Guru, D.S., Manjunatha, K.S., Manjunath, S., Somashekara, M.T.: Interval valued symbolic representation of writer dependent features for online signature verification. Expert Syst. Appl. **80**, 232–243 (2017)
7. Lumini, A., Nanni, L.: Ensemble of on-line signature matchers based on overcomplete feature generation. Expert Syst. Appl. **36**, 5291–5296 (2009)
8. Manjunatha, K.S., Manjunath, S., Guru, D.S., Somashekara, M.T.: Online signature verification based on writer dependent features and classifiers. Pattern Recogn. Lett. **80**, 129–136 (2016)
9. Massone, M., Gabrielli, F., Rineiski, A.: A genetic algorithm for multigroup energy structure search. Ann. Nucl. Energy **105**, 369–387 (2017)
10. Nanni, L.: An advanced multi-matcher method for on-line signature verification featuring global features and tokenised random numbers. Neurocomputing **69**, 2402–2406 (2006)
11. Nanni, L., Lumini, A.: Ensemble of Parzen window classifiers for on-line signature verification. Neurocomputing **68**, 217–224 (2005)
12. Nanni, L., Lumini, A.: Advanced methods for two-class problem formulation for on-line signature verification. Neurocomputing **69**, 854–857 (2006)
13. Ortega-Garcia, J., Fierrez-Aguilar, J., Simon, D., Gonzalez, J., Faundez-Zanuy, M., Espinosa, V., Satue, A., Hernaez, I., Igarza, J.-J., Vivaracho, C., Escudero, D., Moro, Q.-I.: MCYT baseline corpus: a bimodal biometric database. IEE Proc. Vis. Image Sign. Process. **150**, 395–401 (2003)
14. Rutkowski, L.: Computational Intelligence. Springer, Heidelberg (2008)
15. Rutkowski, L., Zalasiński, M., Cpałka, K.: Fuzzy-genetic approach to identity verification using a handwritten signature. In: Advances in Data Analysis and Systems Modeling with Computational Intelligence Methods, vol. 7092. Springer (2017). (in press)
16. Yang, Ch.H., Moi, S.H., Lin, Y.D., Chuang, L.Y.: Genetic algorithm combined with a local search method for identifying susceptibility genes. J. Artif. Intell. Soft Comput. Res. **6**, 203–212 (2016)
17. Yeung, D.Y., Chang, H., Xiong, Y., George, S., Kashi, R., Matsumoto, T., Rigoll, G.: SVC2004: first international signature verification competition. In: Proceedings of the International Conference on Biometric Authentication, pp. 16–22 (2004)

18. Zalasiński, M.: New algorithm for on-line signature verification using characteristic global features. Advances in Intelligent Systems and Computing, vol. 432, pp. 137–142 (2016)
19. Zalasiński, M., Cpałka, K.: New algorithm for on-line signature verification using characteristic hybrid partitions. Advances in Intelligent Systems and Computing, vol. 432, pp. 147–157 (2016)
20. Zalasiński, M., Cpałka, K., Er, M.J.: New method for dynamic signature verification using hybrid partitioning. Lecture Notes In Computer Science, vol. 8468, pp. 216–230 (2014)
21. Zalasiński, M., Cpałka, K., Hayashi, Y.: New method for dynamic signature verification based on global features. Lecture Notes in Computer Science, vol. 8468, pp. 231–245 (2014)
22. Zalasiński, M., Cpałka, K., Hayashi, Y.: A new approach to the dynamic signature verification aimed at minimizing the number of global features. Lecture Notes in Computer Science, vol. 9693, pp. 218–231 (2016)

Knowledge Discovery and Data Mining

Multivariate Regression Tree for Pattern-Based Forecasting Time Series with Multiple Seasonal Cycles

Grzegorz Dudek(✉)

Department of Electrical Engineering, Czestochowa University of Technology, Czestochowa, Poland
dudek@el.pcz.czest.pl

Abstract. Multivariate regression tree methodology is used for forecasting time series with multiple seasonal cycles. Unlike typical regression trees, which generate only one output, multivariate approach generates many outputs in the same time, which represent the forecasts for subsequent time-points. In the proposed approach a time series is represented by patterns of seasonal cycles, which simplifies the forecasting problem and allows the forecasting model to capture multiple seasonal cycles, trend and nonstationarity. In application example the proposed model is applied to forecasting electrical load of power system. Its performance is compared with some alternative models such as CART, ARIMA and exponential smoothing. Application examples confirm good properties of the model and its high accuracy.

Keywords: Multivariate regression tree · Seasonal time series forecasting · Pattern-based forecasting

1 Introduction

Multivariate or multi-output regression aims to simultaneously predict multiple output variables. There are two general approaches for solving multi-output regression problems: either by decomposing the problem into multiple single-output problems or, by adapting a model so that it directly handles multi-output data. The former solution can be time consuming task especially in the case of many outputs and large size of the training data. Moreover, the relationships among the output variables are ignored because they are predicted independently, which may affect the accuracy of the predictions. The latter solution is not as popular because of the complexity of the multivariate regression model. It should be able to capture not only the underlying relationships between input and output variables but also the internal relationships between output variables. According to past empirical works [1, 2] multi-output models ensure better predictive performance especially when the output variables are correlated.

The multivariate regression model learns from the training set $\Psi = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}$, where $\mathbf{x} \in X = \mathbb{R}^n$ is a n-dimensional input vector and $\mathbf{y} \in Y = \mathbb{R}^m$ is a m-dimensional output vector, a target function f which assigns to each input vector an output vector: $f: X \rightarrow Y$. State-of-the-art multi-output regression methods are defined as extensions of standard single-output methods such as statistical methods, support vector machines, kernel methods, regression trees, and classification rules [3].

In the case of forecasting problems the input vector $\mathbf{x}$ includes predictors (past and present data) and the output vector $\mathbf{y}$ includes forecasted variables (future data). The output vector can contains variables of different types, which are dependent on the same predictors, e.g. demand for m various goods, or a single variable in different time-points, e.g. demand for a single good at time-points $t + 1, t + 2, \ldots, t + m$, where m is a forecast horizon. In the experimental part of this work we consider an example of short-term load forecasting, i.e. electrical hourly demand forecasting for the next day. Input vector includes power system hourly loads for the day preceding the forecasted day ($n = 24$) and output vector includes hourly loads for the next day ($m = 24$). Load time series exhibit multiple seasonal cycles of different lengths: daily, weekly and annual (Fig. 1). The daily and weekly profiles change during the year. The daily profile also depends on the day of the week. The load time series expresses trend and is nonstationary in mean and variance. To deal with these all features the load time series is preprocessed to filter out a trend, weekly and annual cycles. The way of time series representation is described in Sect. 2.

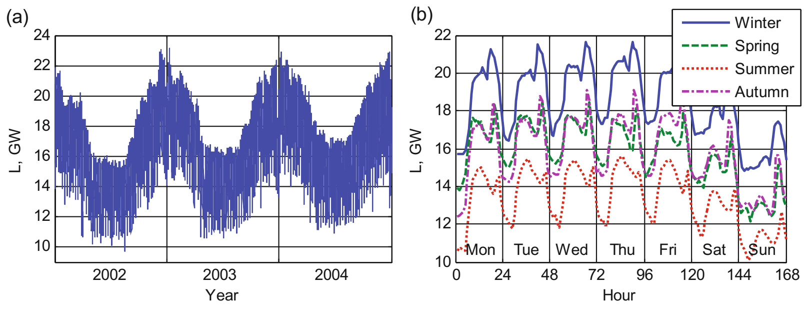


Fig. 1. The hourly electricity demand in Poland in three-year (a) and one-week (b) intervals.

Many various methods has been developed for load time series forecasting. They can be roughly classified as conventional and unconventional ones. Conventional approaches employ regression methods, smoothing techniques and statistical analysis such as ARIMA and exponential smoothing. Unconventional approaches use computational intelligence and machine learning methods such as: neural networks (NN), fuzzy inference systems, neuro-fuzzy systems and support vector machines. These approaches are reviewed in [4, 5]. Most of them are single-output models, but others are able to predict many outputs simultaneously. They include [6–8]: radial basis function NN, generalized regression NN, counterpropagation NN, self-organizing maps, artificial immune systems, nonparametric kernel regression, nearest neighbour regression and clustering-based models.

Regression trees as universal regression methods have been used for load time series forecasting as well [9]. They were originally developed as single-output models but they can be easily adapted to multi-output models. Advantages of multi-output regression trees (MRT) are: much smaller size than a set of single-output regression trees constructed for individual outputs, and taking into account the relationships between outputs when the tree is constructed. In this paper we define and explore MRT for power system load forecasting using patterns of daily cycles as inputs and outputs.

2 Time Series Representations Using Patterns

The daily periods of the load time series are preprocessed and are used as the input vectors for the forecasting model. They express daily shapes or profiles of the time series called x-patterns. The i-th daily period of the load time series $\mathbf{L}_i = [L_{i,1}\ L_{i,2}\ \ldots\ L_{i,n}]$ is represented by the input pattern $\mathbf{x}_i = [x_{i,1}\ x_{i,2} \ldots x_{i,n}] \in X = \mathbb{R}^n$. It is a normalized version of the load vector $\mathbf{L}_i$. The components of x-pattern are defined as follows [10]:

$$x_{i,t} = \frac{L_{i,t} - \bar{L}_i}{D_i} \tag{1}$$

where: $i = 1, 2, \ldots, N$ is the daily period number, $t = 1, 2, \ldots, n$ is the time series element number in the period i, $L_{i,t}$ is the t-th load time series element in the period i, $\bar{L}_i$ is the mean load in the period i, and $D_i = \sqrt{\sum_{l=1}^{n} (L_{i,l} - \bar{L}_i)^2}$ is the dispersion of the time series elements in the period i.

As normalized vectors x-patterns have unity length, zero mean and the same variance. Note that the load time series, which is nonstationary in mean and variance, is represented by x-patterns having the same mean and variance. The x-pattern carries information about the shape of the daily load curve. A trend and also weekly and annual variations are filtered.

An output vector $\mathbf{y}$ expresses the forecasted daily profile. When the forecast horizon is τ (in days), the forecasted load vector $\mathbf{L}_{i+\tau} = [L_{i+\tau,1}\ L_{i+\tau,2} \ldots L_{i+\tau,n}]$ is represented by an output pattern $\mathbf{y}_i = [y_{i,1}\ y_{i,2}\ \ldots\ y_{i,n}] \in Y = \mathbb{R}^n$. Their components are defined as follows:

$$y_{i,t} = \frac{L_{i+\tau,t} - \bar{L}_i}{D_i} \tag{2}$$

where: $i = 1, 2, \ldots, N$, $t = 1, 2, \ldots, n$.

Note that in (2) we use the coding variables $\bar{L}_i$ and D_i for the day i instead of the day $i + \tau$. This is because their values for the day $i + \tau$ are unknown at the moment of forecasting. Using their known values for the day i we can calculate the forecasted load value from transformed Eq. (2):

$$\widehat{L}_{i+\tau,t} = \widehat{y}_{i,t} D_i + \bar{L}_i \tag{3}$$

where $\widehat{y}_{i,t}$ is the t-th component of the y-pattern forecasted by the model.

Due to preprocessing daily periods of the load time series using x- and y-patterns we unify the input and output data and simplify relationships between them. This is further discussed in [10].

3 Multivariate Regression Trees for Forecasting

Multivariate regression trees is an extension of CART (Classification and Regression Trees [11]) proposed by Segal [12]. It works exactly the same way as classical CART, except that there is multiple response variables instead of one. As in CART, the response variables can be continuous or class symbols. Predictor variables can be of different types: quantitative or qualitative. Due to dealing with different types of data, decision trees are an attractive tool for classification and regression problems.

Tree-based methods partition the feature space X into a set of rectangles, and then fit a simple model, usually a constant, in each one. Let us consider multivariable regression problem with multi-input and multi-output continuous variables: $\mathbf{x}$ and $\mathbf{y}$, respectively (in our case $\mathbf{x}$ are n-dimensional input patterns and $\mathbf{y}$ are n-dimensional output patterns). MRT is constructed with a standard top-down induction algorithm. We are starting to build a tree splitting the X space into two regions: $x_j \leq t$ and $x_j > t$. This occurs in node 1 (root node). The variable x_j and split-point t are chosen to achieve the best fit. Then we model the response by the mean of $\mathbf{y}$ in each region. In the next step one or both of regions are split into two more regions in the nodes at the next level of the tree, and this process is continued, until some stopping rule is applied. The result is a partition into the K disjoint regions. The corresponding regression model predicts $\mathbf{y}$ with a constant $\widehat{\mathbf{y}}_k$ in region R_k:

$$\widehat{\mathbf{y}}_k = \frac{1}{N_k} \sum_{i:\mathbf{x}_i \in R_k} \mathbf{y}_i \tag{4}$$

This model is represented by the binary tree. The full dataset of N samples is split into two subsets in the root node. Samples satisfying the condition $x_j \leq t$ at each node are assigned to the left branch, and the others to the right branch. The terminal nodes (leaves) of the tree correspond to the final K regions.

The algorithm of growing a regression tree needs to automatically decide on the splitting variable x_j and split-point t at each node. We seek the splitting variable j among all n variables and split-point t with a greedy algorithm testing all variables and all possible split-points for them. To do so we define a function which is maximized during the searching process. It bases on within node sum of squares (it plays a role of the impurity measure of the node):

$$SS_k = \sum_{i:\mathbf{x}_i \in R_k} \sum_{l=1}^{n} (y_{i,l} - \bar{y}_l)^2 \tag{5}$$

where $\bar{y}_l$ is the average of l-th components of the samples in node k.

The splitting function expressing reduction in impurity after split is of the form:

$$\phi_k(j,t) = SS_k - SS_l - SS_m \tag{6}$$

and the best split maximizes this function:

$$\max_{\substack{j \in \{1,2,\ldots,n\} \\ t \in \Phi_j}} \phi_k(j,t) \tag{7}$$

where l and m are daughter nodes of the node k, which is split, and Φ_j denotes the set of all split-points possible for variable x_j.

The tree is constructed by recursively splitting nodes so as to maximize the above criterion. Tree size is a tuning parameter governing complexity of the model. The optimal tree size should be adaptively chosen from the data. We consider two approaches which determine the tree size. The first one splits tree nodes if the value of criterion (4) is positive (reduction of impurity) and the number of samples in the node is higher than L. This approach produces a tree with internal nodes having at least L samples and leaves having less than L samples. The second approach splits nodes if the impurity is reduced and the variance of y-patterns in the node is higher than v:

$$Var_k = \frac{1}{N_k n} \sum_{i:\mathbf{x}_i \in \Phi_k} \sum_{l=1}^{n} (y_{i,l} - \bar{y}_l)^2 > v \tag{8}$$

Parameters L and v determine the tree size. Higher values of these parameters leads to the smaller trees, which tend to underfitting. On the other hand, too small values lead to bigger trees and overfitting. The optimal values of L and v are estimated in the local version of leave-one-out cross-validation, which is presented in Fig. 2.

```
1. Find $k$ nearest neighbors of the query pattern $\mathbf{x}$ in the
   training set.
2. Build MRT using the training set excluding $i$-th nearest
   neighbor.
3. Calculate MRT error for $i$-th nearest neighbor.
4. Repeat steps 2 and 3 $k$ times for the next nearest neigh-
   bors.
5. Calculate average error over $k$ nearest neighbors as an
   estimate of generalization error.
```

Fig. 2. Algorithm of the local leave-one-out.

4 Application Example

In this section the proposed MRT model was applied to forecast the electricity load demand. As it was described in Sect. 1 electricity load time series exhibits multiple seasonal cycles. The data used for the experiments were retrieved from Polish power system. They contain hourly electricity load data from 2002 to 2004 (the load time series is shown in Fig. 1). The forecast horizon is $\tau = 1$, i.e. 24 h ahead. The MRT models are constructed for 31 days of January 2004 and 31days of July 2004 (in total 62 models for 62 forecasting tasks). For each forecasting task (test sample) the learning set is created individually from the historical data. It contains pairs of patterns representing the same days of the week (Monday,…, Sunday) as the query pattern **x** and forecasted pattern **y**.

Two variants of MRT models are used, which differ in the node splitting criterion:

- MRT1, where the node is split if the impurity is reduced and the number of samples in the node is higher than L,
- MRT2, where the node is split if the impurity is reduced and the variance in y-patterns in the node is higher than v.

The model parameters: number of samples L or variance v, are estimated in the local leave-one-out procedure, in which the validation samples are chosen one by one from the set of $k = 10$ nearest neighbors of the query pattern. In these procedures parameters were estimated using a grid search: L was searched from 4 to 50 with a step size of 2, and v was searched from 0.001 to 0.005 with a step size of 0.0002.

In Fig. 3 an example of MRT1 is shown: tree constructed for July 1, 2004. The optimal value of L was 32 in this case. As can be seen from this figure the tree has fifteen nodes in total, of which eight are leaves. The training y-patterns included in leaves are shown in Fig. 4. Note differences in shape between these patterns. They all represent Thursdays from history (from the period of January 1, 2002–June 30, 2004). Thick lines in this figure express average y-patterns, i.e. the tree response. Details of the tree in Table 1 are shown. There are only six variables used in the node tests: 1, 17, 19, 20, 21 and 22. They are the most important variables among 24 ones in partitioning the

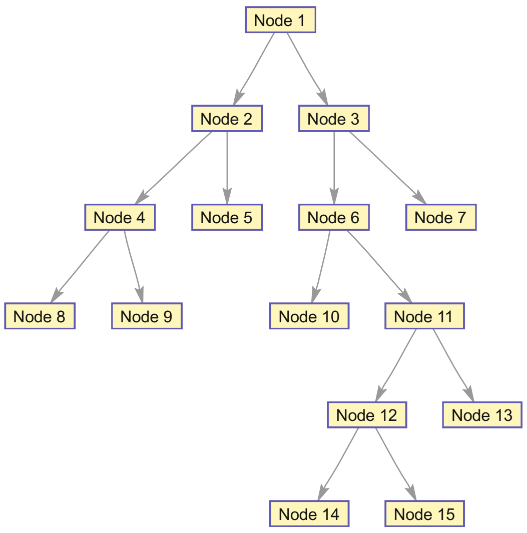


Fig. 3. MRT1 built for July 1, 2004.

input space X into regions. The largest reduction in impurity occurred in the root node (5.58), and also in node 3 (5.36). In the root node the set of 122 samples (the whole training set) was split into two subsets of 55 and 67 samples, respectively. In node 3 the largest reduction in impurity was achieved by excluding only one sample. As we can see form Fig. 4 this is an outlier sample. It represents January 2, 2002. It is atypical because the y-pattern for this day is encoded using the mean load $\bar{L}_i$ and dispersion D_i for the previous day (see (2)), which is New Year's Day. The lower mean load for New Year's Day causes rising of the y-pattern for the next day and consequently its atypical character.

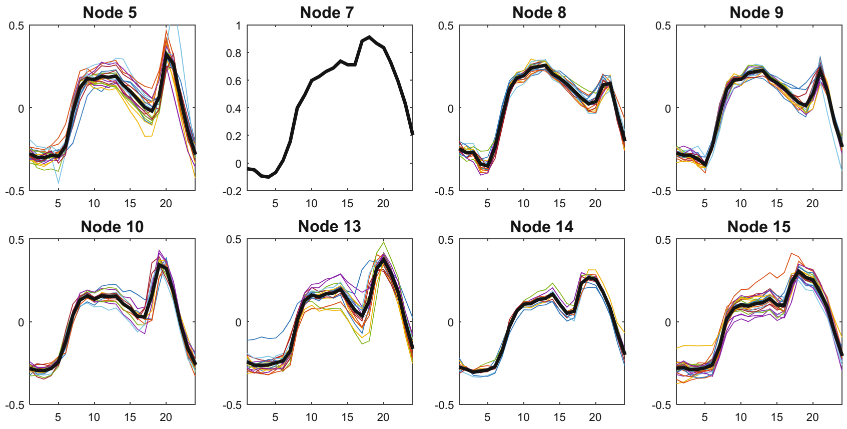


Fig. 4. Y-patterns in terminal nodes of MRT1 built for July 1, 2004; average y-patterns (tree response) with thick lines.

Trees constructed when using MRT2 strategy usually are bigger than trees constructed when using MRT1 strategy. This is shown in Fig. 5 presenting histograms of the number of nodes in the trees for these two cases. The average number of nodes in MRT1 is about 24 and in MRT2 about 36. For July 1, 2004 MRT2 built a tree having 49 nodes, of which 25 are leaves. Optimal value of v in this case was 0.0012. As in MTR1 the largest reduction in impurity (5.58) occurred in the root node, where the training set was split into two subsets of 55 and 67 samples. And also in node 3 (5.36), where the outlier representing January 2, 2002 was separated.

Variables selected for splitting in nodes are shown in Fig. 6. The most often selected variables are: 1, 7, 19 and 20, and the least often selected ones are: 15, 14 and 13.

For comparison the forecasts were performed using ARIMA, exponential smoothing, and classical regression tree (single-output CART). To find the optimal ARIMA and ES models the automated procedures were used implemented in the forecast package for the R system [13]. Distributions of the forecast errors (percentage errors PE) for the five models compared in Fig. 7 are shown. As we can see from this figure for the tree-based models the similar results were obtained. ARIMA and ES generate higher errors. The medians of PE were: 1.29 for ARIMA, 1.25 for ES, 1.01 for

Table 1. MRT1 details for July 1, 2004 (leaves in bold).

Node	Splitting variable j	Split-point t	Impurity reduction $\phi(j,t)$	#samples	Variance of y-patterns *Var*
1	19	0.1927	5.58	122	7.55E−03
2	20	0.1996	1.65	55	3.46E−03
3	1	−0.0686	5.36	67	7.43E−03
4	21	0.1684	0.21	34	1.37E−03
5	**–**	**–**	**–**	**21**	**3.57E−03**
6	22	0.0395	1.64	66	4.17E−03
7	–	–	–	1	0
8	**–**	**–**	**–**	**16**	**1.04E−03**
9	**–**	**–**	**–**	**18**	**1.17E−03**
10	**–**	**–**	**–**	**14**	**1.76E−03**
11	22	0.1284	1.40	52	3.50E−03
12	17	0.1364	0.29	34	1.84E−03
13	**–**	**–**	**–**	**18**	**3.41E−03**
14	**–**	**–**	**–**	**9**	**9.40E−04**
15	**–**	**–**	**–**	**25**	**1.68E−03**

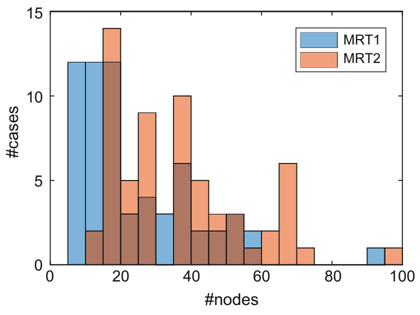


Fig. 5. Histograms of the number of nodes in the trees.

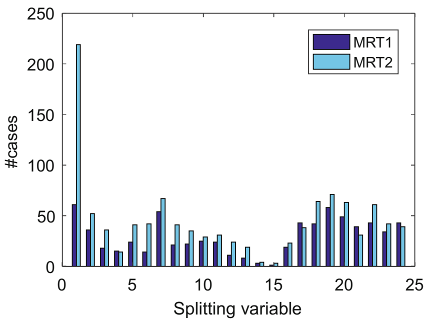


Fig. 6. Splitting variables in the trees.

regression tree, 1.04 for MRT1, and 1.09 for MRT2. The Wilcoxon signed-rank tests indicated that there is no statistically significant difference in errors between regression tree, MRT1 and MRT2. But there is a significant difference in errors between ARIMA/ES and each of tree-based models.

The proposed MRTs generate multi-output response, keeping the relationships between the output variables (y-pattern components). In the case of single-output models, like classical CART, these relationships are ignored because variables are

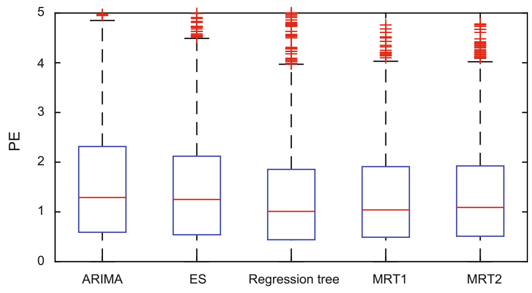


Fig. 7. Boxplots for percentage errors generated by the models.

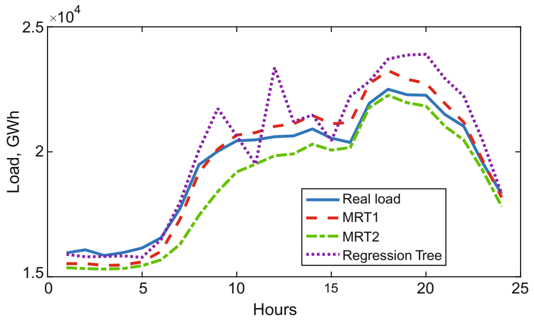


Fig. 8. Real load and forecasts for January 5, 2004.

predicted independently. This may cause lack of smoothness in the forecasted curve. Such example in Fig. 8 is illustrated. For January 5, 2004, in the case of regression tree model we can observe zigzag effect in the middle part of the curve.

5 Conclusion

The proposed model based on multivariate regression trees generates forecasts of the time series many steps ahead in the same time. The model is constructed taking into account not only the underlying relationships between input and output variables but also the internal relationships between output variables. This can lead to improvement in predictive performance especially when the output variables are correlated. The multi-output tree is much smaller than a set of single-output regression trees constructed for individual outputs. So, the learning process is easier and less time consuming. Another advantage of multivariate approach is that the zigzag problem, which appears when single-output model is used for forecasting output variables independently, is eliminated.

In the proposed approach a time series is represented by patterns of seasonal cycles, which simplifies the forecasting problem and allows the model to capture multiple seasonal cycles, trend and nonstationarity. Additional time series transformations such as decomposition, detrending or differencing are not needed.

In the application example the proposed model was applied to forecasting electrical load of power system. It provided as accurate forecasts as classical regression tree and outperformed ARIMA and exponential smoothing models.

References

1. Breiman, L., Friedman, J.H.: Predicting multivariate responses in multiple linear regression. J. R. Stat. Soc. Ser. B **59**(1), 3–54 (1997)
2. Evgeniou, T., Micchelli, C.A., Pontil, M.: Learning multiple tasks with kernel methods. J. Mach. Learn. Res. **6**, 615–637 (2005)
3. Borchani, H., Varando, G., Bielza, C., Larrañaga, P.: A survey on multi-output regression. Wiley Interdisciplinary Rev. Data Mining Knowl. Discov. **5**(5), 216–233 (2015)
4. Tzafestas, S., Tzafestas, E.: Computational intelligence techniques for short-term electric load forecasting. J. Intell. Robot. Syst. **31**, 7–68 (2001)
5. Metaxiotis, K., Kagiannas, A., Askounis, D., Psarras, J.: Artificial intelligence in short term electric load forecasting: a state-of-the-art survey for the researcher. Energy Convers. Manage. **44**, 1525–1534 (2003)
6. Dudek, G.: Neural networks for pattern-based short-term load forecasting: a comparative study. Neurocomputing **2015**, 64–74 (2016)
7. Dudek, G.: Pattern similarity-based methods for short-term load forecasting – Part 2: models. Appl. Soft Comput. **36**, 422–441 (2015)
8. Dudek, G.: Artificial immune system with local feature selection for short-term load forecasting. IEEE Trans. Evol. Comput. **21**(1), 116–130 (2017)
9. Dudek, G.: Short-term load forecasting using random forests. In: Filev, D., et al. (eds.) Intelligent Systems 2014, Advances in Intelligent Systems and Computing, vol. 323, pp. 821–828 (2015)
10. Dudek, G.: Pattern similarity-based methods for short-term load forecasting – Part 1: principles. Appl. Soft Comput. **37**, 277–287 (2015)
11. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Chapman & Hall, New York (1984)
12. Segal, M.R.: Tree-structured methods for longitudinal data. J. Am. Stat. Assoc. **87**(418), 407–418 (1992)
13. Hyndman, R.J., Khandakar, Y.: Automatic time series forecasting: the forecast package for R. J. Stat. Softw. **27**(3), 1–22 (2008)

Active Protocol Discoverer Based on Grammatical Evolution

Dariusz Pałka[1(✉)], Marek Zachara[1], and Krzysztof Wójcik[2]

[1] AGH University of Science and Technology, Mickiewicza 30, 30-059 Kraków, Poland
{dpalka,mzachara}@agh.edu.pl

[2] Tadeusz Kościuszko Cracow University of Technology, Jana Pawła II 37 Avenue, 31-864 Kraków, Poland
krzysztof.wojcik@mech.pk.edu.pl

Abstract. The paper presents a proposition of a system of protocol discovering (Protocol Discoverer) developed on the basis of Grammatical Evolution techniques. Unlike numerous other solutions based solely on observing messages between participants of a conversation, our Protocol Discoverer is an active participant which generates messages and sends them to the system for which the protocol is to be identified. This solution allows not only for identifying typical behaviors of participants within a protocol, but also for finding anomalous behaviors (the ones which normally do not occur between participants using a defined protocol).

In order to generate the description of a protocol in the form of a context-free grammar, the solution presented in the article is based on the evolutionary approach using Grammatical Evolution by Grammatical Evolution. Universal grammar is the basis for creating evolutionary solution grammars which describe particular pairs of requests-responses appearing in the protocol.

Keywords: Protocol discovering · Grammatical evolution

1 Introduction

Protocol discovering is a process of finding some form of a description of a protocol on the basis of samples of conversations between the participants of these conversations.

Protocol discovering has numerous applications, which include discovering models of a software process [1], inferring business protocols of web services [2], workflow mining in Enterprise Resource Planning (ERP), systems such as SAP, Customer Relationship Management (CRM) software, Business to Business (B2B) applications [3], discovering communication protocols used by hardware devices, inferring application-level protocols in order to detect potential vulnerabilities [4], etc.

The discovered protocol can be described in various forms, for example, deterministic and nondeterministic finite state machines (FSM), push-down automata, Petri nets, regular grammars, and context-free grammars.

There are several methods of discovering an unknown protocol, such as: data mining techniques [5], using neural nets to identify FSM [6, 7], or using RNET, KTAIL and MARKOV methods [1].

These techniques and their limitations in protocol discovery are strongly connected with grammar inference, i.e. knowing positive samples of the language (sentences in this language) and, optionally, negative samples of the language (sentences not possible in this language), it is possible to infer a grammar that represents this language. Theoretical aspects of the limitations of the process of grammar inference were widely studied by Gold [8, 9].

An important conclusion of these studies states that finding even deterministic finite state automata (which are equivalent to regular grammars, i.e. the simplest class of grammar in Chomsky's hierarchy) with a minimum number of states, given positive and negative samples of the language, is NP-hard and it is impossible given only positive samples of the language [8, 9].

Most protocol discovery methods are based on passive protocol discovering, i.e. systems which only observe conversations between participants and do not generate any messages [1, 2, 4, 10].

In the paper we propose and discuss an active Protocol Discoverer, i.e. a system which takes an active part in conversations by generating requests to the other participant of the conversation and tries to discover the protocol that is used by this participant.

The grammar of the protocol will be inferred in an evolutionary way using Grammatical Evolution by Grammatical Evolution [11].

2 Meta-Grammar Genetic Algorithm for Protocol Discovering

In our study we aim at discovering a bipartite protocol, i.e. a protocol that covers an active conversation between two participants. It is assumed that each conversation consists of consecutive pairs of request-response. Additionally, we take into consideration only situations in which the request is always sent by the same participant of the conversation. This situation is analogous to communication between a client and a server, where a client is an active participant of the conversation and initiates it.

The objective of the study is to propose a mechanism which automatically generates requests (imitates an active participant of the conversation) and, on the basis of responses received from the other participant, creates a description of the protocol used by this participant.

To describe the protocol, context-free grammar is used. This formalism is capable of describing the majority of protocols currently in use, as most specifications of known protocols use Backus–Naur form of context-free grammar or regular expressions/ grammars (which are subsets of context-free grammars).

In the process of grammar inference describing a protocol an evolutionary approach based on Grammatical Evolution (GE) [12, 13] is used. This process has numerous applications, e.g. it allows for discovering some untypical web service behaviors that can indicate possible web service vulnerabilities [14]. Because the solution sought by the evolutionary process in this application is a grammar, a variant of the GE, called the

Grammatical Evolution by Grammatical Evolution $(GE)^2$, [11] is used, in which the universal (meta) grammar dictates the construction of the solution grammar (grammar describing a discovered protocol). Generating a solution grammar in an evolutionary way on the basis of a meta grammar has many advantages, for example, it allows for finding a proper grammar that can later be used in grammar-based Genetic Programming [15].

2.1 Meta Grammar

A meta grammar guides the evolutionary process of finding solution grammars i.e. grammars that best describe the protocol being discovered.

In the study it is assumed that requests and responses in a communication protocol may be expressed by any byte string[1], so any context-free grammar (in Backus–Naur form) that generates binary strings can be a solution grammar. A meta grammar should make it possible to generate such solution grammars. A proposed meta grammar is presented below.

```
<Grammar>    ::= <Rule>
             |   <Grammar> <Rule>
<Rule>       ::= <RuleName> <Assign> <List> <LineEnd>
<RuleName>   ::= <NT1> | <NT2> | <NT3> | <NT4>
             |   <NT5> | <NT6> | <NT7> | <NT8>
<Assign>     ::= "::="
<List>       ::= <Term>
             |   <Term> <WhiteSpace> <List>
<Term>       ::= <RuleName>
             | <Byte>
<Byte>       ::= "0" | "1" | "2" | "3"
             |   "4" | "5" | "6" | "7"
             |   "8" | "9" | "A" | "B"
             |   "C" | "D" | "E"| "F"
             |   "10" | "11" | "12" | "13"
    ...
             | "FC" | "FD" | "FE" | "FF"
<WhiteSpace>  ::= " "
<LineEnd>    ::= "\n"
```

This meta grammar allows for generating solution grammars which contain up to eight nonterminal symbols (NT1, NT2, …, NT8). The number of nonterminal symbols is one of the parameters of the evolution.

An example of a solution grammar generated by $(GE)^2$ with the use of the meta grammar presented above is:

[1] A byte string is a sequence of one or more bytes. The text string is a special case of a byte string, so protocols that use text messages may be inferred by the proposed mechanism.

```
<NT1> ::= 1C <NT1>
        |   F5 FC 2B <NT5> A8
        |   A5 <NT1> <NT8>
<NT5> ::= 49 4E
        |   51
<NT6> ::= 87
<NT7> ::= <NT1>
        |   B0
```

3 Protocol Decomposition

In the presented approach we assume that a conversation between two participants consists of consecutive pairs of requests-response. One participant of the conversation is active, i.e. sends requests, and the other is passive, i.e. sends responses to the received requests. Moreover, the protocol is assumed to be stateless, which means that a response to a particular request does not depend on previous requests; the possibility of enhancing protocol discovery by removing this restriction is discussed in the last section.

The system should be able to discover the protocol used by the passive participant of the conversation, i.e. the form of correct requests and corresponding responses. Because the passive participant of the conversation is treated as a 'black box', which means that there is no extra information about the results of the conversation, (e.g. in a semantic form), all knowledge about this system is based on the syntax of requests and corresponding responses.

The idea proposed in the study employs an evolutionary approach with Grammatical Evolution (precisely, its GE^2 variant) to infer a grammar describing the protocol being discovered. However, instead of looking for a complete grammar describing the whole protocol (all correct requests and corresponding responses), the proposed system infers separate grammars for each unique request-response pair.

Generally, the protocol may be designed in such a way that many different requests trigger the same response. This can happen, for example, if requests differ only by certain parameters, and responses do not depend on these parameters. Since it is assumed that all our knowledge about the protocol is based only on responses for generated requests (the passive participant of the conversation is a 'black box'), all requests generating the same response are treated as equivalent. Therefore, the whole protocol is clustered into subprotocols for request-response pairs with the same response. For each subprotocol the grammar is inferred (this grammar is called Request-Response Grammar or RRGrammar) using GE^2 technique, and, finally, the grammar for whole protocol is created from grammars of subprotocols (which is discussed later).

A general form of RRGrammar is as follows:

```
<RRGrammar> ::= <Request> <Response>
<Request>   ::= // grammar for the request is inferred
            // using GE
<Response>  ::= // a particular response
            //(a sequence of bytes)
```

where:

<Request> is a nonterminal symbol defined by the grammar inferred for the subprotocol. On the right hand side (RHS) of the production for <Request> there is a start symbol of the grammar inferred for requests for the subprotocol.

<Response> is a particular byte string (a sequence of bytes). As mentioned above, this byte string is unique for each subprotocol grammar.

The sequence diagram presenting the process of creating the grammar for the discovered protocol is shown in Fig. 1.

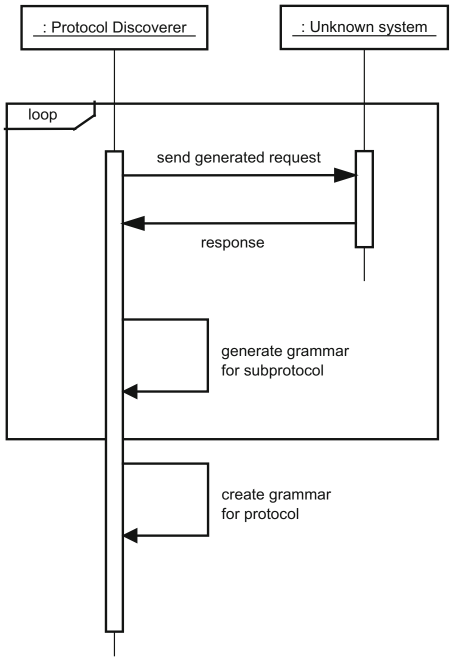


Fig. 1. The sequence diagram presenting interactions between Protocol Discoverer and the system with an unknown protocol.

4 The Process of Protocol Discovering

Applying the method of protocol decomposition and Grammatical Evolution by Grammatical Evolution technique presented above, the proposed evolutionary process of protocol discovering can be described as follows:

1. The initial generation of individuals (solution grammars) is created using a universal (meta) grammar
2. For each solution grammar the pool of sentences (byte strings) is randomly generated. Each generated sentence represents a request which is sent to the system with an unknown protocol (the passive participant of the conversation)
3. Each generated request is sent to the system with an unknown protocol and the response from the system is registered
4. If the response is new (never seen before), a new cluster for the subprotocol is created. Such a cluster contains a separate population of individuals (solution grammars) which represents the language of the request for this subprotocol. Each cluster evolves independently of other clusters and the best grammar describing a given request-response pair is identified. When a new cluster is created, the solution grammar which has generated the request triggering a new response is added to the population of individuals
5. On the bases of requests generated by it and the responses received, the value of a fitness function is calculated for each solution grammar
6. On the basis of fitness values, candidates (solution grammars) for the next generation are selected within each cluster
7. The next generation of individuals is created for each cluster separately using genetic operations (such as crossover and mutation).
8. The process loops to step 2, and co-evolution of each cluster continues until the stop condition occurs
9. The grammar for the whole protocol is created as a composition of grammars for particular subprotocols

4.1 Stop Conditions of the Evolutionary Process

There is no natural stop condition in the process of protocol discovering presented above. In a general scenario the context-free grammar (or even regular grammar) cannot be inferred using a finite set of samples (without extra knowledge about the language). So, the transition from GE loop (step 8 in the description of the process above and the loop fragment in Fig. 1) to creating the grammar for the whole protocol must be triggered by artificial stop conditions, such as the maximum number of generations or exceeding the maximum time allowed for the process, etc. Of course, it cannot be guaranteed that the description of the protocol obtained this way will be complete. After generating the grammar for the whole protocol (step 9), the process can return to the loop of the evolutionary algorithm in order to further improve the description of the discovered protocol.

4.2 Fitness Function

A crucial element of the described evolutionary process of protocol discovering is the method of evaluation of particular individuals (solution grammars), which is traditionally named the fitness function. The fitness function is directly used for selecting individuals (RRGramars) which best represent the subprotocol within clusters.

The fitness function for grammar G is defined as in Eq. 1:

$$fitness(G) = w1 \cdot \sum\nolimits_{s \in S} b(s) + w2 \cdot \sum\nolimits_{s \in S} (1 - b(s)) + w3 \cdot \sum\nolimits_{s \in S} p(s) \tag{1}$$

where :

S is a set of all (distinct) sentences (requests) generated from grammar G in a given generation

b(s) – is the membership function of sentences *s* to a given cluster. It is defined as:

$$b(s) = \begin{cases} 1 \mathit{if\ request\ s\ belongs\ to\ the\ cluster} \\ 0\ \mathit{otherwise} \end{cases} \tag{2}$$

Request *s* belongs to a particular cluster if the response to this request is the same as defined for the cluster.

C is a set of all known (generated before and stored) requests that belong to a given cluster

p(s) is defined as follows:

$$p(s) = \begin{cases} 1\ \mathit{if\ s\ is\ parsable\ by\ grammar\ G} \\ -1\ \mathit{otherwise} \end{cases} \tag{3}$$

w1, w2, w3 – are scaling constants

The fitness function in the form described above prefers individuals (solution grammars) which can generate the greatest number of requests belonging to a given cluster, the smallest number of requests not belonging to a given cluster[2], and is able to parse the greatest number of requests known before belonging to a given cluster.

4.3 Genetic Operators

In this study genetic operators of mutation, crossover, duplication and pruning typical for GE [12, 13] were used.

5 Preliminary Results

In this section the preliminary test results of the Protocol Discoverer are presented.

[2] With the assumption that the scaling constant w1 is greater than w2.

5.1 Testing Environment

Preliminary tests were conducted to discover the control protocol used in a Pan-Tilt-Zoom camera.

Camera VISCA was chosen for testing Protocol Discoverer.

VISCA is a control protocol designed by Sony [16, 17] and used by many tele-conference and surveillance cameras. This protocol uses RS-232 protocol as Data Link Layer.

The testing environment is presented in Fig. 2

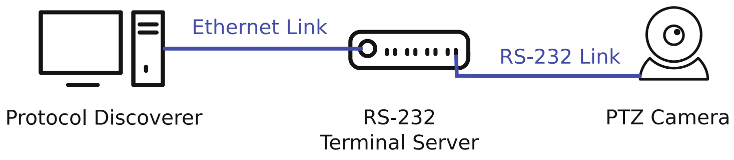


Fig. 2. Test environment

PTZ camera was linked to a computer with running Protocol Discoverer through RS-232 Terminal Server. In the future such a solution can allow for simultaneous connection of numerous appliances (cameras) for which a protocol is being discovered. This will increase the speed of the discovery process, as the bottle neck in this process is the waiting time needed for sending the request and receiving the response.

5.2 Process Parameters

The parameters of the process of protocol discovering used in preliminary tests are listed in Table 1.

Table 1. Evolution parameters used in tests

Parameter	Value
Max number of nonterminal symbols in solution grammar	8
Population size for each request-response cluster	100
Tournament size	5
Number of sentences (byte strings) generated in each generation from solution grammar (max no of elements in set S from Eq. 1)	100
Fitness function scaling factor w1	2
Fitness function scaling factor w2	1
Fitness function scaling factor w3	1

5.3 Results

The results for the first 15 generations in the evolutionary process of protocol discovering are presented below. The results obtained for protocol discovering with random requests (byte strings) and sequential requests were used as comparative values for assessing efficiency of the proposed process of protocol discovering.

Random requests were created by generating the same number of requests as generated in the evolutionary process, and the length of the request and the value of particular bytes were generated in a random way with a uniform distribution. The maximum length of the request generated in a random way corresponded to the maximum length of the request generated in a given generation by an evolutionary algorithm.

Sequential requests were created by generating consecutive requests in the form of byte strings in the lexicographical order (that is [0], [1], [2], … [255], [0,0], [0,1], [0,2], … [0,255], [1,0], [1, 1],…). The number of requests corresponded to their number generated by an evolutionary algorithm in given generations.

Table 2. The comparison of results obtained by Protocol Discoverer employing Grammatical Evolution ("GE") techniques presented in the paper, random requests, ("Random") and sequential requests ("Sequence")

Generation no	Number of requests	Max length of request	No of clusters		
			GE	Random	Sequence
1	40	6	1	1	1
2	40	6	1	1	1
3	98	3	1	1	1
4	91	2	1	1	1
5	97	2	1	1	1
6	103	28	1	1	1
7	119	37	1	1	1
8	195	40	1	1	1
9	459	43	2	1	1
10	591	34	2	2	1
11	661	535	6	2	1
12	848	979	8	2	1
13	1376	955	20	2	1
14	2365	1015	44	3	1
15	5135	1015	105	3	1

The first cluster found by all techniques (GE, Random and Sequence) contained an empty response (no response in a set timeout). It is the most frequent behavior of devices with implemented VISCA protocol (it occurs when the structure of the request is erroneous).

As can be seen in Table 2, both the presented Protocol Discoverer (GE) and random requests yielded another cluster (the first without an empty response) after almost the same number of generated requests (GE in generation 9 and Random in generation 10).

In the process of further search the proposed Protocol Discoverer based on GE turned out to be much more efficient – it found 105 clusters (i.e. requests generating various responses) in the first 15 generations, while generating requests in a random way yielded only 3 various responses (clusters). It results from the fact that the process of searching for new requests based on the structure (grammar) of previously found requests is much more effective (at least for the analyzed VISCA protocol) than a random search. Generally, messages for a given protocol have similar syntax, thus, finding syntax for one of them facilitates finding syntaxes for the other ones. In the presented evolutionary method crossover and mutation operators turned out to be crucial.

Creating sequential requests generated only an empty response, which was connected with the fact that for a set number of generated requests in 15 generations (there were 12218 requests altogether) only one- and two-byte requests were created. According to the VISCA specification [16, 17], there are no correct requests (commands) shorter than three bytes.

Examples of responses discovered by Protocol Discoverer

The evolutionary method of protocol discovering for a device with VISCA protocol yielded requests generating correct (consistent with protocol specification) responses, such as:

[90, 60, 41, FF][3] - where: 90 denotes the sender's address 1 and the receiver's address 0 (controller); 60, 41- error message "command not executable"; FF - response terminator.

Responses not consistent with the specification included a two-byte sequence **[C6, FF]** generated to the request [88, AA, F7, A8, AA, F7, A8, AA, F7, A8, FF], which is also not defined in the VISCA specification.

As can be seen in this example, the proposed mechanism allows for finding not only requests that result in correct responses and responses defined in the specification of the VISCA protocol, but also requests which generate responses not documented previously.

6 Conclusions

Protocol discovering is useful in many situations in which - on the basis of interactions between participants - we want to find a general description of a set of rules used by these participants.

The paper presents active Protocol Discoverer, i.e. a discoverer which not only creates the description of the observed protocol but also generates messages (an active participant of the interaction), which allows it to obtain knowledge on the protocol used by the other participant (whose protocol is unknown). To describe the protocol, context-free grammar is employed, which is created in an evolutionary way on the bases of $(GE)^2$ techniques and clusters of the responses.

The proposed approach assumes that communication between two participants within the protocol is conducted through exchanging pairs of requests-responses,

[3] Presented values are given in the hexadecimal system.

including an empty response (the lack of response) as a specific example of the response. An additional assumption is that the protocol is stateless, which means that requests sent earlier do not influence responses to the requests sent later. This restriction can be partially mitigated by modifying the proposed scheme of protocol discovering in such a way that the pairs of requests-responses are not considered individually but in groups "request$_1$-response$_1$, request$_2$-response$_2$,…,request$_n$-response$_n$" and treat the series "request$_1$-response$_1$,…request$_n$" as one request and "response$_n$" as the response to this request. Groups should be selected in such a way that requests from one group would not influence the responses of another group (i.e. the protocol should be stateless from the point of view of the whole groups). Such approach significantly increases the complexity of the whole process of evolution and we intend to test its usefulness and efficiency in further studies.

Preliminary tests prove that the proposed technique is more effective in discovering requests generating various responses than a random search (or sequential generation of all possible requests). Additionally, even for a device with a relatively simple protocol like VISCA, the proposed Protocol Discoverer can find requests (commands) triggering responses incompatible (or not documented) in the specification of this protocol. The results of preliminary tests imply that the method of protocol discovering presented in the paper can also be used in examining incompatibilities of implementing the protocol with the specification and in discovering potential vulnerabilities of the protocol to attack attempts.

References

1. Cook, J.E., Wolf, A.L.: Discovering models of software process from even-base data. ACM Trans. Softw. Eng. Methodol. **7**, 215–249 (1998)
2. Saint-Paul, R., Casati, F., Motahari-Nezhad, H.R., Benatallah, B.: Protocol discovery from imperfect service interaction logs. In: IEEE 29th International Conference on Data Engineering (ICDE), pp. 1405–1409 (2007), doi:10.1109/ICDE.2007.369022
3. van der Aalst, W.M.P., van Dongen, B.F., Herbst, J., Maruster, L., Schimm, G., Weijters, A. J.M.M.: Workflow mining: a survey of issues and approaches. Data Knowl. Eng. **47**, 237–267 (2003)
4. Cui, W., Kannan, J., Wang, H.J.: Discoverer: automatic protocol reverse engineering from network traces. In: Provos, N. (ed.) USENIX Security Symposium. USENIX Association (2007)
5. Fayyad, U., Uthurusamy, R.: Data mining and knowledge discovery in databases. Commun. ACM **39**(11), 24–36 (1996)
6. Das, S., Mozer, M.C.: A unified gradient-descent/clustering architecture for finite state machine induction. In: Proceedings of the 1993 Conference, vol. 6. Advances in Neural Information Processing Systems, pp. 19–26. Morgan Kaufmann (1994)
7. Zeng, Z., Goodman, R.M., Smyth, P.: Learning finite state machines with self-clustering recurrent networks. Neural Comput. **5**, 976–990 (1993)
8. Gold, E.M.: Language identification in the limit. Inf. Control **10**, 447–474 (1967)
9. Gold, E.M.: Complexity of automatic identification from given data. Inf. Control **37**, 302–320 (1978)

10. Comparetti, P.M., Wondracek, G., Kruegel, Ch., Kirda, E.: Prospex: Protocol Specification Extraction. In: IEEE Symposium on Security and Privacy, pp. 110–125. IEEE Computer Society (2009)
11. O'Neill, M., Ryan, C.: Grammatical evolution by grammatical evolution: the evolution of grammar and genetic code. In: Keijzer, M., O'Reilly, U.M., Lucas, S.M., Costa, E., Soule, T. (eds.) Genetic Programming 7th European Conference, EuroGP 2004, Proceedings. LNCS, vol. 3003, 5–7 April, pp. 138–149. Springer, Portugal (2004)
12. O'Neill, M., Ryan, C.: Grammatical evolution. IEEE Trans. Evol. Comput. **5**(4), 349358 (2001). doi:10.1109/4235.942529
13. O'Neill, M., Ryan, C.: Grammatical evolution: evolutionary automatic programming in a arbitrary language. Genetic Programming, vol. 4. Kluwer Academic Publishers (2003)
14. Pałka, D., Zachara, M., Wójcik, K.: Evolutionary scanner of web application vulnerabilities. In: Gaj, P., Kwiecien, A., Stera, P. (eds.) CN, Communications in Computer and Information Science, vol. 608, pp. 384–396. Springer (2016), http://dx.doi.org/10.1007/978-3-319-39207-3_33
15. Pałka, D., Zachara, M.: Automatic grammar induction for grammar based genetic programming. In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC (1). Lecture Notes in Computer Science, vol. 9119, pp. 350–360. Springer (2015), http://dx.doi.org/10.1007/978-3-319-19324-3_32
16. Sony EVI-D30/D31 Command List, https://www.cs.rochester.edu/~nelson/courses/vision/resources/sony_evi-d31.pdf, Accessed 16 May 2017
17. Sony Color Video Camera Technical Manual, https://pro.sony.com/bbsccms/assets/files/mkt/remotemonitoring/manuals/rm-EVID100_technical_manual.pdf, Accessed 16 May 2017

Speaker Diarization Using Deep Recurrent Convolutional Neural Networks for Speaker Embeddings

Pawel Cyrta[1(✉)], Tomasz Trzciński[1,2], and Wojciech Stokowiec[1,3]

[1] Tooploox, Warsaw, Poland
pawel.cyrta@tooploox.com, t.trzcinski@ii.pw.edu.pl, wojciech.stokowiec@pjwstk.edu.pl
[2] Warsaw University of Technology, Warsaw, Poland
[3] Polish-Japanese Academy of Information Technology, Warsaw, Poland

Abstract. In this paper we propose a new method of speaker diarization that employs a deep learning architecture to learn speaker embeddings. In contrast to the traditional approaches that build their speaker embeddings using manually hand-crafted spectral features, we propose to train for this purpose a recurrent convolutional neural network applied directly on magnitude spectrograms. To compare our approach with the state of the art, we collect and release for the public an additional dataset of over 6 h of fully annotated broadcast material. The results of our evaluation on the new dataset and three other benchmark datasets show that our proposed method significantly outperforms the competitors and reduces diarization error rate by a large margin of over 30% with respect to the baseline.

Keywords: Speaker diarization · Speaker embeddings · Speaker clustering · Deep neural network · Recursive convolutional neural networks · Convolutional neural networks

1 First Section

Speaker diarization [1] aims at splitting an audio signal into homogeneous segments according to the speaker identity. More precisely, the goal of speaker diarization is to answer the question of "who spoke when" within a given audio stream. Its application is required in many real-life applications related closely to information retrieval, such as speech-to-text transcription [2] or speaker recognition [3]. Use case scenarios of speaker diarization techniques include the analysis of phone conversations, political debates or lectures and conferences, where determining how many people speak and when they are active is crucial for the proper understanding of information.

Although several advancements in developing high-quality speaker diarization algorithms have been made in the recent years [4–8], there are still many challenges that have to be addressed, e.g. analysis of the overlapping speech or speaker's voice modulations. A typical approach to solving those challenges is to split speaker diarization problem into three parts: voice activation detection ("speech/no speech"), speaker change detection ("same/different speaker") and speaker identification ("speaker A/B/C"). This partitioning allows to simplify speaker diarization problem, nevertheless it introduces several sub-optimal objectives to be optimized across different modules.

This paper address these shortcomings by learning a set of speaker embeddings that can be used throughout the diarization process. We propose learning the embeddings by training recurrent convolutional neural network for a speaker classification task and prove that this leads to good generalization of the embeddings, even for the speakers not included in the training dataset. Moreover, instead of using manually hand-crafted spectral features such as MFCC [9], PLP [10] or RASTA [11], we apply a recurrent neural network directly on the magnitude spectrograms (SFTF) to learn a set of high-level feature representations also referred to as *speaker embeddings*. Our motivation to use SFTF and not raw audio stream stems from the fact that deep architectures were successfully used to learn Mel-like filters from power spectrum [12], while applying them directly on raw audio data did not lead to classification performance improvement [13–15]. Inspired by recent advancements in the speaker diarization domain achieved with convolutional neural networks (CNNs) [16] and successful applications of recurrent convolutional neural networks (R-CNNs) to other problems, such as image classification [17] or birds' sound classification [18], we propose to employ a recurrent neural network architecture to speaker diarization problem. Our motivation to use this approach is based on the observation that the temporal patterns present in audio streams can be better aggregated and interpreted with recurrent architectures, due to the specific feedback embedded in their design.

To show the superiority of our method over the state of the art, we evaluate it on an exhaustive set of three publicly available datasets with over 720 h of recording and over 1400 speakers. Additionally, we collect a new dataset of fully annotated audio stream of broadcast news material and release it to the public. As far as we know, this is the only publicly available broadcast dataset. We use this dataset to extend the comparison of our method with the competitive approaches and show that speaker embeddings trained on other datasets generalize well to the new dataset. Our evaluation results clearly show that our approach significantly outperforms the state-of-the-art methods by reducing a classification error rate by a large margin of over 30%.

To summarize, the contributions of this paper are threefold:

- We bridge the gap between the raw audio signal and the output of speaker diarization system using deep neural network architecture.
- To the best of our knowledge, this is the first attempt to use recurrent convolutional neural networks to model low-level speaker embeddings.
- Finally, we introduce a new fully labeled dataset of news, interviews and debates that contain over 6 h of recording that can empower future research in the domain of speaker diarization.

The remainder of this paper is organized in the following manner: Primarily, we discuss related work and state of the art methods. We then present our method along with the input features and deep neural network architecture. Subsequently, we describe our evaluation setup along with the datasets used for experiments. Finally, we show the results of our evaluation along with the conclusions.

2 Related Work

Over the recent years speaker diarization systems have been successfully used to analyse human speech in various everyday scenarios - from phone calls [19] through business meetings [20] to broadcast news [7]. In this paper, we mainly focus on the last use case, i.e. the goal of our speaker diarization methods is to automatically annotate broadcast TV and radio streams discussing current news and events.

Historically, the research done in the domain of speaker diarization was highly influenced by the methods proposed for speaker verification. Most notable example is the i-vector framework [7], currently considered as a standard speaker recognition system[1]. An i-vector is an information-rich low-dimensional vector extracted from a feature sequence that represents a speech segment, sometimes known as an *audio voice-print* per analogy to fingerprints. The i-vectors are typically computed on the MFCC features [9] which represent both speaker and channel features. For the proper performance of the speaker diarization system based on i-vectors, it is therefore necessary to add a disambiguation phase which relies on PLDA [21] - a clustering method - and the resulting clustering scores. Since the quality of clustering highly depends on the size of the analysed segments, processing short segments of speech with not enough information leads to significant performance drop of the whole system [22].

To address the above-mentioned shortcomings of the i-vector frameworks, several techniques of so-called anchor modeling were introduced [23]. They create a representation of speech utterance from a set of pre-trained speaker models in order to get a likelihood score of each anchor. Those representations are then grouped into characterization vectors that describe individual speakers. The anchor modeling allows to embed the information about the speaker into the speech segment representation and is often referred to as a *speaker embedding* model. Similar embeddings have recently been obtained through deep neural network training [24]. More precisely, a neural network with three hidden layers was applied to Gaussian Mixture Universal Background Model (GMM-UBM) of MFCC features with the goal of speaker classification. The activations of the so-trained neural network were then used as features of previously unseen speakers. Since then, several extensions of this architecture were proposed, including additional LSTM units [25] and temporal-pooling layers [19, 26]. Once again, the activations of the resulting network were used as speaker embeddings.

In this paper we draw an inspiration from these works and propose a deep neural network architecture to create speaker embeddings for speaker diarization system. Contrary to the previous works, we postulate using recurrent convolutional neural networks as our feature extractor. The motivation to use this architecture stems from the fact that recurrent convolutional neural networks were successfully employed in similar acoustic modeling applications [27] and show state-of-the-art performances on many other related classification tasks, such as document [28], image [17], bird songs [18] and music genre classification [29]. Furthermore, we decided to use CQT-grams instead of raw audio signal, as an input to our recurrent convolutional neural network.

[1] http://voicebiometry.org/.

We motivate this decision by the inferior performances obtained for deep architectures applied on the raw signals [15, 30].

3 Method

In this section we first define speaker diarization problem and show its relation to speaker embeddings. We then introduce a set of features used as inputs of our method and follow up with a brief description of the proposed deep neural network architecture.

3.1 Problem Formulation

The goal of a speaker diarization system is to analyse an audio stream and output a set of labels defining the moments when each individual speaker speaks. This can be cast as a classification task, if all the speakers along with their identities are known beforehand. In real-life applications, however, this is rarely the case. We therefore address the speaker diarization problem using a two-step approach. First, we train a neural network in a supervised manner with a goal of speaker classification. Then, we use the pre-trained neural network to extract time-dependent speaker characteristics, the so-called speaker embeddings. More precisely, we use activations from the last layer of neural network as speaker embeddings. We aggregate the sigmoid outputs by summing all outputs class-wise over the whole audio excerpt to obtain a total amount of activation for each entry and then normalizing the values by dividing them with the maximum value among classes. The analysis of those embeddings in time allows the system to detect speaker change and identify the newly appearing speakers by comparing the extracted and normalized embedding with those previously seen. If the cosine similarity metric between the embeddings is higher than a threshold, fixed at 0.4 after a set of preliminary experiments, the speaker is considered as new. Otherwise, we map its identity to the one corresponding to the nearest embedding.

Since the first stage of our approach involves training a neural network for a speaker classification task, we formulate here the training objective of the network. The input of our network is a weighted spectrogram and a set of ground truth labels defining speaker identity. We train the network to minimize the cross-entropy of the predicted and true distributions:

$$L(y, \hat{y}) = -\sum_{i=1}^{N} \sum_{j=1}^{C} y_i^j \log\left(\hat{y}_i^j\right), \quad (1)$$

where denotes the ground-truth label of sample *i* and refers to the prediction probability estimated by the network; *N* and *C* indicate the number of training samples and classes respectively.

3.2 Features

As mentioned above, the input of our neural network architecture is a weighted spectrogram. More precisely, we use basic time-frequency representation, i.e. a magnitude spectrogram (SFTF), with some perceptual weighting filter banks applied on the spectrogram to increase the method robustness to noise and pitch variability. Although several previous works reported worse performances when using deep architectures with SFTF instead of Mel-spectral features [29, 31], our preliminary results showed that weighting SFTF with proper perceptual weighting filters may overcome those shortcomings. Below we describe the weighting filters that were separately used to test as input to network in our experiments:

- log-Mel: We transform an input spectrogram by applying Short-Term Fourier Transform and compute Mel filter bank features [9] on the spectral output. To that end, we warp the frequency axis to match the Mel frequency scale. The resulting output is a 96-dimensional binned distribution.
- Gammatone: We apply a Gammatone filter [32] to an input spectrogram to obtain Gammatone-grams. Gammatone filter can be interpreted as an approximative model of the sound filtering performed by human hearing system. The resulting output is again a 96-dimensional representation.
- CQT: We apply a Constant Q Transform [33] an input spectrogram to generate the so-called CQT spectrograms (or CQT-grams). The transform is computed on four octaves with 24 bins per octave and minimum frequency of 80 Hz. The resulting output is also a 96-dimensional vector.

Before applying the filter banks on the spectrograms, we apply several preprocessing steps that aim at normalizing the input signal. First, we transform a stereo audio signal to mono by averaging over two channels. We take fixed-length audio signal segments of 3.072 s (96 frames of 512 audio samples) as an input to the system. Each segment is extracted every 250 ms, as this sampling frequency was reported to improve clustering [21, 26]. We then downsample the signal to 16 kHz in order to grasp only the frequency range that is most relevant to speaker identification. Afterwards we apply a Hamming window of 512 samples (32 ms) with hop of 256 samples combined with a pre-emphasis filter and Fast Fourier Transform (FFT). All the parameters of the preprocessing steps were selected based on a set of preliminary experiments.

3.3 Network Architecture

In this section we present the architecture of the proposed recurrent convolutional neural network (R-CNN) which is trained to model speaker embeddings.

Our network consists of 11 learnable layers: 4 convolutional blocks followed by 2 recurrent layers with 1 fully-connected at the end. Each convolutional block consists of a convolutional layer, a batch-normalization layer [34] and ELU nonlinearity, as well as a max-pooling layer.

We select hyperparameters of the convolutional blocks such as the sizes of convolution kernels and a max-pooling operator, based on a set of preliminary results. For each convolutional block we use the same size of convolutional kernel: 3×3.

As far as max-pooling layers are concerned, following kernel sizes have been used (2×2) in the first block, (3×3) in the second, then (4×4) in the third and in the fourth. For max-pooling layer we use stride equal to kernel width to make those operations non-overlapping. We treat the resulting feature map of size $N \times 1 \times 15$ as a sequence of 15 N-dimensional vectors and feed them in recurrent layers with GRU gating mechanism [35].

Our preliminary results suggest that employing exponential linear units (ELUs) introduced in [36] instead of ReLUs slightly improves classification accuracy. Contrary to the claim made in [36], we did not observe any speed-ups in learning. During training we observed an internal covariate shift, i.e. the change in distribution of each layer's inputs resulting from the updates of parameters of the previous layers. This typically slows down the training process and requires lower learning rates along with a careful parameter initialization [34]. To address these problems, we extended our architecture with an additional batch-normalization layer placed after each convolutional layer.

We regularize the network with a dropout rate [37] of 0.1 after each convolutional block, and a stronger rate of 0.3 after the recurrent block. We choose dropout rates based on the preliminary results. To minimize the expression form Eq. 1 we employ Stochastic Gradient Descent with mini-batches of size 128 with ADAM optimization algorithm [38].

4 Experiments

In this section we first introduce datasets used to train and evaluate our method. We then describe implementation details of our method and the baselines. Finally, we present the evaluation metrics along with the results of our experiments.

4.1 Datasets

To evaluate our method and compare it with the state of the art, we use following publicly available datasets: AMI meeting corpus [39] (100 h, 150 speakers), ISCI meeting corpus [40] (72 h, 50 speakers), and YouTube (YT) speakers corpus [41] (550 h, 998 speakers). Those datasets cover a wide range of different types of voices of English speakers and various recording quality standards.

To extend our evaluation method, we collected a set of broadcast material from major news stations: CNN, MSNBC, Fox News, Bloomberg, RT America, BBC. For each of the channels we acquired 1 h of YouTube clips with 3.2 individual speakers on average per clip and 36.4 speakers per channel. The average speech material for an individual speaker is 2 min and 12 s. The resulting material is over 6 h long and contains various types of news programs, interviews and debates. It was then manually labeled with speakers names assigned to every speech segment. Since the dataset also contains pointers to videos, it can also be used for speech recognition, face detection and optical character recognition tasks. We release our Broadcast News Videos dataset

(BNV) to the public[2]. One shall note that due to the license restrictions on the video material, the dataset contains only tools and annotations and not raw video files. To recreate a complete dataset, one shall use the download scripts available in the repository.

The overview of the datasets can be seen in Table 1. In total, the datasets used for evaluation cover over 1400 h of recording and over 720 individual speakers. Furthermore, for evaluation purposes all datasets except for the Broadcast News Video dataset are randomly split into training and testing sets with the proportion of 70% to 30%. Broadcast News Video dataset is used only for testing since its goal is to verify the performance of our speaker diarization method on previously unseen material. When evaluating the methods on Broadcast News Video dataset, the training is done on a union of all the other datasets, that is on AMI, ISCI and YouTube speakers corpus.

Table 1. Overview of the datasets used in the experiments. All datasets contain 728 h of recordings and 1416 speakers.

Dataset	# speakers	#hours
AMI [39]	150	100
ISCI [40]	50	72
YouTubeSpeakersCorpus (YT) [41]	998	550
Broadcast News Videos (BNV)[2]	218	6

4.2 Implementation

We implement our method that uses recurrent convolutional neural network in Python. For acoustic feature extraction, we employ Yaafe toolkit[3] and we derive magnitude, Mel and CQT spectrogram using this software. To build a deep neural network architecture we use Keras [42], together with Theano [43]. Our model was trained on Nvidia Titan X GPU and the training took 18 h.

4.3 Baseline

The baseline for our comparison is a state-of-the-art LIUM Speaker Diarization system [44]. LIUM is based on a GMM classifier and uses 13 MFCC audio features as an input. The features are calculated using frames of 25 ms with a Hamming window and 10 ms overlap. It also relies on CLR clustering of speech segments into speakers.

Additionally, we compare the performance of our method with both convolutional neural network (CNN) and recurrent convolutional neural network (R-CNN) that takes magnitude spectrogram (SFTF) as an input. The goal of this comparison is to validate our hypothesis that using filter banks improves the results with respect to the systems that rely on magnitude spectrograms.

[2] http://github.com/cyrta/broadcast-news-videos-dataset.

[3] http://yaafe.sourceforge.net/.

4.4 Evaluation Metric

As our performance evaluation metric we use Diarization Error Rate (DER) [1]. This metric takes into account both segmentation and classification errors, as well as errors from speech activity detection stage. The Diarization Error Rate is computed as:

$$DER = E_{Spk} + E_{FA} + E_{Miss}, \tag{2}$$

where E_{Spk} is speaker error, defined as the time assigned to incorrect speakers divided by total time, E_{FA} refers to false alarm speech, defined as amount of time incorrectly detected as speech divided by total time and E_{MISS} refers to miss speech, defined as the amount of speech time that has not been detected as speech divided by total time. The definition of DER also includes acceptance margin of 250 ms which compensate for human errors in reference annotation.

4.5 Results

The results of the evaluation can be seen in Table 2. Our proposed deep learning architecture based on recurrent convolutional neural network and applied to CQT-grams outperforms the other methods across all datasets with a large margin. Its improvement reaches over 30% with respect to the baseline LIUM speaker diarization method with default set of parameters.

Table 2. Diarization Error Rates obtained for all datasets. Our proposed R-CNN architecture with CQT-gram features clearly outperforms the other methods. The performance improvement over the baseline LIUM method reaches over 30%.

Method	Features	Datasets			
		AMI	ISCI	YT	BNV
LIUM [44]	MFCC	25.1	21.6	26.3	28.5
CNN	SFTF	22.4	20.4	24.2	24.6
	log-Mel	19.2	18.1	21.5	22.3
	Gammatone	18.9	17.5	21.3	22.1
	CQT	16.7	15.3	20.1	21.4
R-CNN	SFTF	21.7	19.3	23.7	24.1
	log-Mel	15.7	14.2	19.1	20.1
	Gammatone	15.6	14.1	18.6	19.7
	CQT	**15.3**	**13.8**	**17.8**	**19.6**

The results of the R-CNN combined with other features, namely log-Mel-grams and Gammatone-grams are on par with the best performing configuration, while significantly outperforming vanilla convolutional neural networks with the same inputs. In fact, all types of input features combined with the R-CNN architecture give on average 12% improvement with respect to the results of the CNN architecture. These results clearly show that the recurrent character of our proposed method allows to better model speaker embeddings and therefore leads to better overall accuracy of the speaker diarization system.

5 Conclusions

In this work we proposed a novel method that learns speaker embeddings for speaker diarization problem using recurrent convolutional neural network architecture. To the best of our knowledge, this is the first approach that employs recurrent convolutional neural networks to model low-level speaker embeddings in the context of speaker diarization. We evaluate our method on three publicly available datasets and a new dataset we collected and released to the public. We prove that by using recurrent deep learning architecture our proposed method is able to outperform the state-of-the-art speaker diarization method by a large margin of over 30% in terms of Diarization Error Rate across all evaluated datasets.

Our future research plans include verifying the effectiveness of deep neural network architectures in the context of noise reduction and source separation. We also plan to investigate unsupervised deep learning techniques, such as deep clustering [45] and recursive autoencoders, on large datasets to obtain speaker embeddings that are able to discriminate between bigger corpora of speakers.

Acknowledgment. The authors would like to thank NowThisMedia Inc. for enabling this research.

References

1. Miro, X.A., Bozonnet, S., Evans, N.W.D., Fredouille, C., Friedland, G., Vinyals, O.: Speaker diarization: a review of recent research. IEEE Trans. Audio Speech Lang. Process. **20**(2), 356–370 (2012)
2. Gupta, V., Kenny, P., Ouellet, P., Stafylakis, T.: I-vector-based speaker adaptation of deep neural networks for French broadcast audio transcription. In: ICASSP (2014)
3. Liu, Y., Tian, Y., He, L., Liu, J.: Investigating various diarization algorithms for speaker in the wild (SITW) speaker recognition challenge. In: Interspeech (2016)
4. Le Lan, G., Meignier, S., Charlet, D., Deleglise, P.: Speaker diarization with unsupervised training framework. In: ICASSP (2016)
5. Woubie, A., Luque, J., Hernando, J.: Short-and long-term speech features for hybrid hmm-i-vector based speaker diarization system. In: Odyssey (2016)
6. Bredin, H., Gelly, G.: Improving speaker diarization of tv series using talking-face detection and clustering. In: ACM on Multimedia Conference (2016)
7. Xu, Y., McLoughlin, I., Song, Y., Wu, K.: Improved i-vector representation for speaker diarization. Circ. Syst. Sig. Process. **35**(9), 3393–3404 (2016)
8. Ferras, M., Madikeri, S., Motlicek, P., Bourlard, H.: Systemfusion and speaker linking for longitudinal diarization of tv shows. In: ICASSP (2016)
9. Mermelstein, P.: Distance measures for speech recognition, psychological and instrumental. Pattern Recog. Artif. Intell. **116**, 374–388 (1976)
10. Hermansky, H.: Perceptual linear predictive (PLP) analysis of speech. J. Acoust. Soc. Am. **87**(4), 1738–1752 (1990)

11. Hermansky, H., Morgan, N.: Rasta processing of speech. IEEE Trans. Audio Speech Lang. Process. **2**(4), 578–589 (1994)
12. Sainath, T.N., Kingsbury, B., Mohamed, A., Ramabhadran, B.: Learning filter banks within a deep neural network framework. In: Workshop on Automatic Speech Recognition and Understanding (2013)
13. Zhu, Z., Engel, J.H., Hannun, A.Y.: Learning multiscale features directly from waveforms. In: Interspeech (2016)
14. Hoshen, Y., Weiss, R.J., Wilson, K.W.: Speecha coustic modeling from raw multi channel waveforms. In: ICASSP (2015)
15. Palaz, D., Magimai-Doss, M., Collobert, R.: Analysis of CNN-based speech recognition system using raw speech as input. In: Interspeech (2015)
16. Lukic, Y., Vogt, C., Dürr, O., Stadelmann, T.: Speaker identification and clustering using convolutional neural networks. In: International Workshop on Machine Learning for Signal Processing (MLSP) (2016)
17. Zuo, Z., Shuai, B., Wang, G., Liu, X., Wang, X., Wang, B., Chen, Y.: Convolutional recurrent neural networks: learning spatial dependencies for image representation. In: CVPR (2015)
18. Cakir, E., Adavanne, S., Parascandolo, G., Drossos, K., Virtanen, T.: Convolutional recurrent neural networks for bird audio detection. In: ICASSP (2017)
19. Snyder, D., Ghahremani, P., Povey, D., Garcia-Romero, D., Carmiel, Y., Khudanpur, S.: Deep neural network-based speaker embeddings for end-to-end speaker verification. In: IEEE Spoken Language Technology Workshop (2016)
20. Yella, S.H.: Speaker diarization of spontaneous meeting room conversations. Ph.D. dissertation, Ecole Polytechnique Federale de Lausanne (2015)
21. Sell, G., Garcia-Romero, D.: Speaker diarization with plda i-vector scoring and unsupervised calibration. In: 2014 IEEE Spoken Language Technology Workshop (2014)
22. Vesnicer, B., Zganec-Gros, J., Dobrisek, S., Struc, V.: Incorporating duration information into i-vector-based speaker recognition systems. In: Odyssey: The Speaker and Language Recognition Workshop, pp. 241–248 (2014)
23. Mami, Y., Charlet, D.: Speaker identification by location in an optimal space of anchor models. In: Interspeech (2002)
24. Rouvier, M., Bousquet, P., Favre, B.: Speaker diarization through speaker embeddings. In: 23rd European Signal Process- ing Conference, EUSIPCO (2015)
25. Bredin, H.: Tristounet: triplet loss for speaker turn embedding. CoRR, abs/1609.04301 (2016)
26. Garcia-Romero, D., Snyder, D., Sell, G., Povey, D., McCree, A.: Speaker diarization using deep neural networks. In: ICASSP (2017)
27. Trigeorgis, G., Ringeval, F., Brueckner, R., Marchi, E., Nicolaou, M.A., Schuller, B., Zafeiriou, S.: Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network. In: ICASSP (2016)
28. Tang, D., Qin, B., Liu, T.: Document modeling with gated recurrent neural network for sentiment classification. In: EMNLP (2015)
29. Choi, K., Fazekas, G., Sandler, M., Cho, K.: Convolutional recurrent neural networks for music classification. arXiv preprint arXiv:1609.04243 (2016)
30. Ghahremani, P., Manohar, V., Povey, D., Khudanpur, S.: Acoustic modelling from the signal domain using CNNs. In: Interspeech 2016 (2016)
31. Dieleman, S., Schrauwen, B.: End-to-end learning for music audio. In: ICASSP (2014)
32. Patterson, R., Nimmo-Smith, I., Holdsworth, J., Rice, P.: An efficient auditory filterbank based on the gammatone function. A meeting of the IOC Speech Group on Auditory Modelling at RSRE, vol. 2(7) (1987)

33. Brown, J.C.: Calculation of a constantq spectral transform. J. Acoust. Soc. Am. **89**(1), 425–434 (1991)
34. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. CoRR, abs/1502.03167 (2015)
35. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. CoRR, abs/1412.3555 (2014)
36. Clevert, D., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (ELUs). CoRR, abs/1511.07289 (2015)
37. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Improving neural networks by preventing co-adaptation of feature detectors. CoRR, abs/1207.0580 (2012)
38. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. CoRR, abs/1412.6980 (2014)
39. Carletta, J., Ashby, S., Bourban, S., Flynn, M., Guillemot, M., Hain, T., Kadlec, J., Karaiskos, V., Kraaij, W., Kronenthal, M., Lathoud, G., Lincoln, M., Lisowska, A., McCowan, I., Post, W., Reidsma, D., Wellner, P.: The ami meeting corpus: a pre-announcement. In: MLMI (2006)
40. Janin, A., Baron, D., Edwards, J., Ellis, D., Gelbart, D., Morgan, N., Peskin, B., Pfau, T., Shriberg, E., Stolcke, A., Wooters, C.: The ICSI meeting corpus, pp. 364–367 (2003)
41. Schmidt, L., Sharifi, M., Moreno, I.L.: Large-scale speaker identification. In ICASSP (2014)
42. Chollet, F.: Keras (2015). https://github.com/fchollet/keras
43. Al-Rfou, R., et. al.: Theano: a python framework for fast computation of mathematical expressions. CoRR, abs/1605.02688 (2016)
44. Meignier, S., Merlin, T.: Lium spkdiarization: an open source toolkit for diarization. In: CMU SPUD Workshop (2010)
45. Hershey, J.R., Chen, Z., Roux, J.L., Watanabe, S.: Deep clustering: discriminative embeddings for segmentation and separation. In: ICASSP (2016)

Classification Tree for Material Defect Detection Using Active Thermography

Grzegorz Dudek[(✉)] and Sebastian Dudzik

Department of Electrical Engineering,
Czestochowa University of Technology, Czestochowa, Poland
{dudek, sebdud}@el.pcz.czest.pl

Abstract. Active thermography is a highly efficient and powerful technique that enables us to detect the subsurface defects by heating the investigated material sample and recording the thermal response using an infrared camera. In this work a simple variant of the time-resolved infrared radiometry method was used. The study was conducted for a sample made of the low thermal diffusivity material with artificially produced aerial defects. As a result of experiment, the sequence of thermograms was obtained. Heating and cooling curves for each thermogram pixel were determined and treated as patterns describing local features of the material. These patterns are recognized by classification tree and classified into two categories: "defect" or "non-defect". Advantages of classification tree is an automatic feature selection and strong reduction of the pattern dimensionality. On the basis of simulation study, it can be concluded that classification tree is a useful tool for the characterisation and detection of material defects.

Keywords: Active thermography · Classification tree · Defect detection

1 Introduction

Non-destructive testing methods are used to detect material discontinuities (i.e. defects) without changing their properties [1]. This enables us to control the manufacture quality of the structural elements of devices, their components and even end products. Typical application areas of the non-destructive testing include:

- aerospace industry,
- automotive industry,
- chemical and petrochemical industries,
- conventional and nuclear power industry,
- railways,
- construction.

In non-destructive testing various methods are used to ensure the quality of products, e.g.: ultrasonic, radiological, eddy current, as well as the passive and active thermography [1, 2]. The presence of defect in subsurface material layer can be stated using a thermal wave theory [3, 4]. This theory plays a fundamental role in the defect detection with the active thermography methods. According to the theory, when we

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 38th International Conference on Information Systems Architecture and Technology – ISAT 2017*, Advances in Intelligent Systems and Computing 655, DOI 10.1007/978-3-319-67220-5_11

subject a material sample to the periodic thermal extortion with amplitude T_{bA}, the temperature value at time τ and at depth z can be described using a thermal wave equation [1]:

$$T(z,\tau) = T_{bA} \cdot \exp\left(-\frac{z}{\mu}\right) \cos\left(\omega \cdot \tau - \frac{2 \cdot \pi \cdot z}{\lambda}\right) \tag{1}$$

where: T_{bA} – amplitude of the periodic thermal excitation, τ – time, z – depth, μ and λ are the thermal diffusion length and thermal wave length, respectively, expressed as [5]:

$$\mu = \sqrt{\frac{2 \cdot \Lambda}{\omega \cdot \rho \cdot c_p}} = \sqrt{\frac{2 \cdot a}{\omega}}, \quad \lambda = 2\pi\mu \tag{2}$$

where: Λ – thermal conductivity of the material, ω – thermal wave frequency, ρ – material density, c_p – material specific heat, a – thermal diffusivity of the material.

The active thermography relies on the heating of the investigated material surface and then recording the temperature field over the time [5–7]. Depending on the experimental procedure, various thermal excitation sources are used. In particular, there is possible to use the short energy pulse (pulsed thermography), periodically variable heating (lock-in thermography) and long pulse with a low energy (stepped heating). In each of these methods, a recording of the temperature field in the heating phase, cooling phase or both of them is conducted. The presence of defect is stated on the basis of the analysis of recorded temperature field. Different methods are used for this case: methods based on the modelling of the heat transfer occurred in the investigated material sample, digital processing methods or machine learning methods [5–8]. As a result of the stepped heating method, data vectors with multiple features (representing temperatures) are obtained. Therefore dimensionality reduction methods (e.g. PCA) are required for classification purposes [8]. In this paper the classification tree is used to analyze the sequence of thermograms. This method is equipped with built-in feature selection mechanism, which is a valuable advantage in this application.

From the point of view of the experimental practice, the lock-in and the pulse methods can be difficult to apply [1]. Therefore, in this work, a simple variant of the time-resolved infrared radiometry method was used [1, 5]. In this variant, the geometry of investigated sample is described by a two-layer model, wherein the second layer, denoted as L2, represents a subsurface defect (Fig. 1).

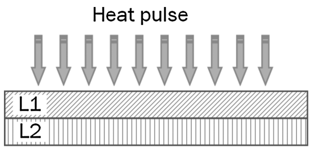


Fig. 1. Two-layer model of the investigated sample.

During the experiment, a temperature rise is monitored over the all excitation period (i.e. the sample under investigation is constantly heated at low power for a specified time). Eventually, as the result of experiments, the thermal image sequence (the sequence of thermograms) is obtained.

In this work, on the basis of sequence of thermograms the heating-cooling curves (H-CCs) are determined and analyzed. The classifier using the classification tree is proposed to recognize "defect" and "non-defect" areas of the sample. The input patterns of the tree are H-CCs.

The remaining sections of the paper are organized as follows. In Sect. 2, experimental investigations using active thermography are described. Section 3 presents a classification tree for defect detection on the basis of thermograms. In Sect. 4, we analyze the thermograms and evaluate the performance of the classification tree. Finally, Sect. 5 is a summary of our conclusions.

2 Experimental Investigations Using Active Thermography

The active thermography experiments were conducted with an experimental setup presented in Fig. 2.

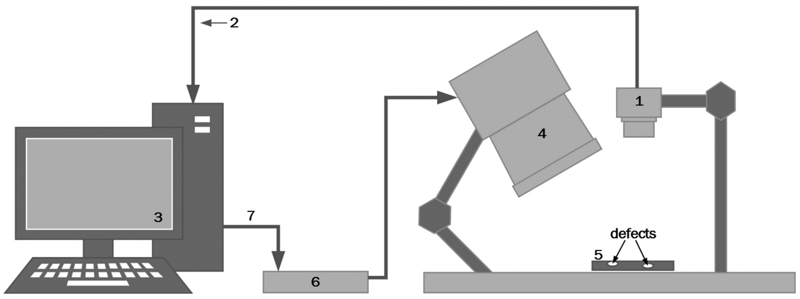


Fig. 2. The experimental setup used in the active thermography investigations.

The setup comprises of:

1. Long-wave infrared camera IRS336-NDT (FPA detector, spatial resolution: 336256 pixels, NETD < 50 mK) used to record the thermal image sequence.
2. Digital interface GigEVision with GenICam used for linking the camera with PC.
3. PC equipped with the software for data acquisition and processing. It cooperates with the heat excitation controller and the infrared camera.
4. Halogen lamp with 2.0 kW power equipped with a power amplifier controlled by a heat excitation controller.
5. Sample under investigation contained artificial defects.
6. Heat excitation controller.
7. USB cable.

The sample under investigation was made of Plexiglas. Nine blind flat-bottomed holes at different depths were drilled in the sample. The holes are made with different diameters ranging from 2.0 mm to 3.4 mm. The thickness of the sample was equal to 10 mm. To avoid the effect of an external radiation, the heated surface of the sample was covered with a paint of high band emissivity coefficient ($\varepsilon \approx 0.95$). A scheme of the investigated sample geometry is shown in Fig. 3.

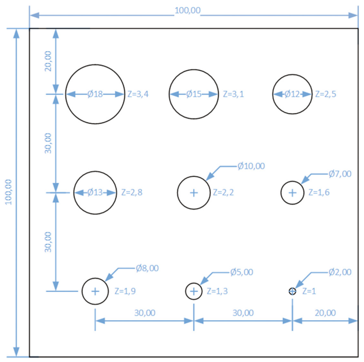


Fig. 3. The scheme of the investigated sample geometry (each dimension in mm)

In the experiments the surface of the sample was heated with a halogen lamp for a 120 s with 70% power. At the same time the sequence of thermal images was recorded using the infrared camera. The recording was conducted at frame rate equals to 5 Hz. Further, the cooling phase lasting 80 s without the lamp excitation was recorded as well. As a result of experiment, the sequence of 1000 thermograms was obtained. Each thermogram consists of 211×212 pixels.

Some examples of thermograms of the surface of investigated sample in Fig. 4 are shown. They were recorded after time $\tau = 20$ s (early heating phase), $\tau = 120$ s (end of the heating phase) and $\tau = 199$ s (late cooling phase). Analysing the results shown in Fig. 4, you can observe the blur of the imaged defects, which is strongly dependent on the time instant of recording. The blur is due to the nature of the heat transfer process occurred in the investigated sample. Inside the sample the heat flows in all directions, including the plane parallel to the surface under investigation. It affects the change in surface temperature distribution. Between defect and non-defect areas the heat flow is disturbed and the transition zones appear. The presence of the zones can cause problems at the data processing stage, especially when the transition pixels should be used as a source for the training data in machine learning routines.

In Fig. 5 the H-CCs are shown measured in two points representing “defect” and “no defect” classes. As can be seen from this figure the curve of class “defect” is above the curve for class “non-defect”. This is because of the lesser local heat capacity in the defect points, where the material layer is thinner. This phenomenon can be used for defect detection. In the next section classification tree is proposed, which classify H-CCs and thus recognizes defect regions.

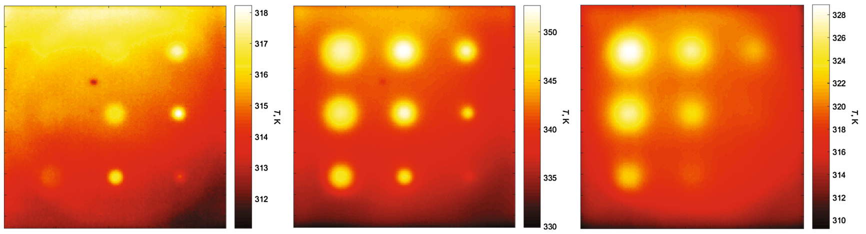


Fig. 4. Thermograms of the surface of investigated sample recorded at τ = 20 s (left), τ = 120 s (middle), and τ = 199 s (right).

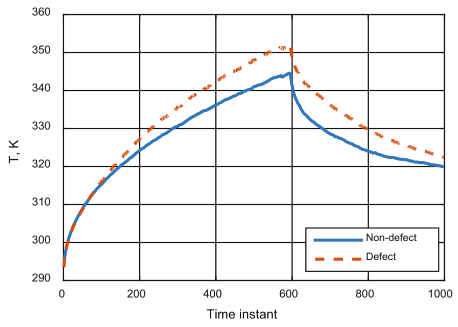


Fig. 5. The heating-cooling curves representing "defect" and "non-defect" classes.

3 Classification Tree for Defect Detection

Decision trees are popular tools of machine learning used for inductive inference. They approximate a target function discreetly and represent it in a tree structure or alternatively in the set of IF-THEN decision rules. The advantage of decision trees over other methods of data classification and function fitting, e.g. neural networks, are their direct interpretability (clear logical rules implemented in the tree) and the ability to act not only on quantitative but also qualitative (nominal and ordinal) variables.

We focus on the variant of decision trees proposed by Breiman [9] called CART (Classification and Regression Trees). In our case CART learns to solve the classification problem, where the response variable y is a class symbol: "defect" or "non-defect". Input patterns are vectors $\mathbf{x} \in \mathbb{R}^n$ representing H-CCs. The successive components of $\mathbf{x}$ are temperatures at successive time instants. Each thermogram pixel is represented by a pair ($\mathbf{x}$, y), where $\mathbf{x} = [T_1\ T_2\ \ldots\ T_n]$ is a H-CC measured for this pixel, and $y \in$ {"defect", "non-defect"}. For N pixels we have a set of N training instances $\Omega = \{(\mathbf{x}_i, y_i), (\mathbf{x}_i, y_i),\ldots, (\mathbf{x}_N, y_N)\}$.

An example of classification tree is shown in Fig. 6. The tree consists of intermediate nodes (marked with triangles), which perform tests on input variables (components of $\mathbf{x}$), terminal nodes (leaves; marked with points) with labels indicating a class, and branches connecting nodes. CART is a binary tree, i.e. each intermediate node has only two children.

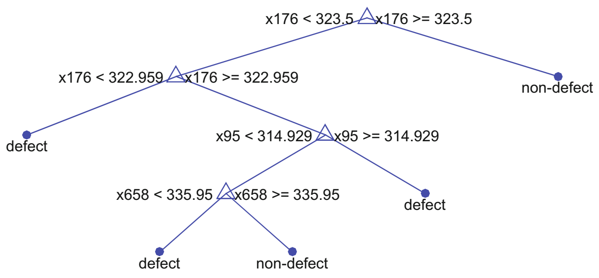


Fig. 6. Example of CART.

The tree is constructed with a top-down induction algorithm (recursive tree-growing process). In the intermediate nodes the set of instances is split into two subsets: positive instances that meet the test assigned to the node, and negative ones that do not meet this test. In the case of continuous variables this is an inequality test of the form:

$$\chi(\mathbf{x}) = \begin{cases} 1, \text{ if } x_i < s_i \\ 0, \text{ if } x_i \geq s_i \end{cases} \tag{3}$$

where x_i is a splitting variable and s_i is a split-point.

The splitting variables and split points are selected using the a greedy algorithm which tests all variables (1000 successive temperature values of the H-CC in our case) and all possible split-points. The best split is selected, which maximizes the drop in impurity of the node defined by:

$$\Delta I_i = I - p_L I_L - (1 - p_L) I_R \tag{4}$$

where I is an impurity of the parent node, I_L and I_R are the impurities of the left and right children nodes, and p_L is the fraction of instances that will go to the left node when test χ is used.

As an impurity measure Gini's diversity index is used, which for two classes is defined as:

$$I = 2q(1 - q) \tag{5}$$

where q is the observed fraction of instances that are in category "defect".

The stopping rule is: the node is pure (it contains only instances of one class) or there is no reduction in impurity or there are fewer than L instances in the node. When a node meets one of these conditions it becomes a leaf. The parameter L controls the depth of the tree. Deeper trees with many leaves are usually highly accurate on the training data, but tend to overfit. They are not guaranteed to show a comparable accuracy on a test set. Shallow trees can be more robust and easy to interpret.

4 Simulation Study

In this section we analyze the thermograms for the sample fragment of 39 × 39 pixels around the middle hole ϕ10. In Fig. 7 the H-CCs recorded for all 1521 pixels are shown. They form two overlapped bands: upper band including curves for defect pixels and lower band including curves for non-defect pixels. Pixels for which H-CCs lie in the overlapping region can be unrecognized. As we can see from Fig. 7 the overlapping band is not very wide. A narrow overlapping band means that the classes are well separated, and that the temperature has a great discriminative power, i.e. the ability to separate classes "defect" and "non-defect". But the width of the overlapping band is dependent on the time. This is shown in Fig. 8 presenting the number of pixels lying in the overlapping region. For lower time instants the classes are poorly recognizable because most of 1521 pixels lie in the overlapping region. For time instants greater than about 100 the number of pixels in overlapping region decreases rapidly to several dozen or even a dozen or so.

In Fig. 9 the thermograms recorded at time instant 50 (τ = 10 s), 100 (τ = 20 s), and 600 (τ = 120 s) are shown. The pixels within the defect area are marked with black dots. Pixels lying in overlapping region are marked with rings. As can be seen from this figure for τ = 10 s pixels from the upper part of the thermogram and pixels from the defect area lie in the overlapping region. Their temperatures are very similar. After a time of τ = 20 s, the differences in temperatures between "defect" and "non-defect" pixels are more distinct. Temperatures in "defect" pixels are higher than in "non-defect" ones. The pixels from the transition zone between defect and non-defect areas lie in the overlapping region. There is 74 of such pixels. After τ = 120 s their number is reduced to 18. Thus, the temperature has the higher discriminative power for τ = 120 s than for 10 and 20 s.

Classification tree constructed for defect detection on the basis of thermograms in Fig. 10 is shown. The training set consisted of 1521 labeled x-patterns representing H-CCs for each pixel. The classification error estimated using 10-fold cross-validation was 0.53%. As we can see from Fig. 10 only three points of the curves were selected as splitting variables: T_{776}, i.e. the temperature recorded after the time τ = 155.2 s, T_{73},

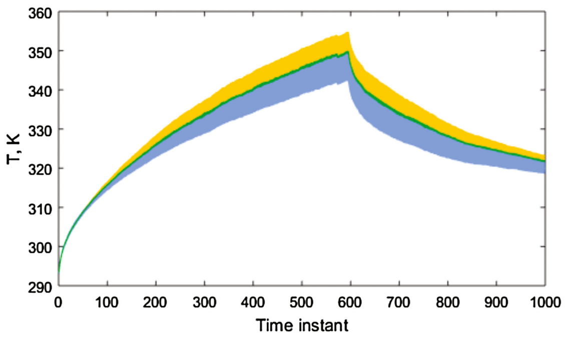


Fig. 7. The heating-cooling curves representing "defect" (upper band) and "non-defect" (lower band) classes for all pixels.

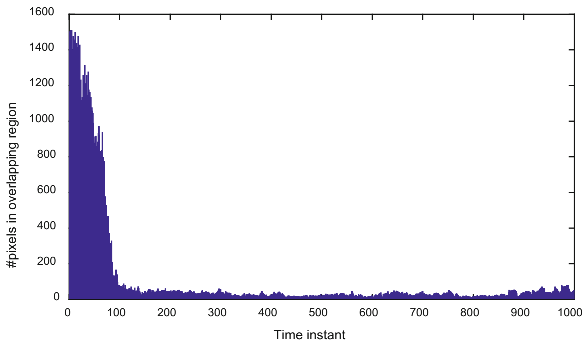


Fig. 8. Number of pixels for which the heating-cooling curves lie in the overlapping band.

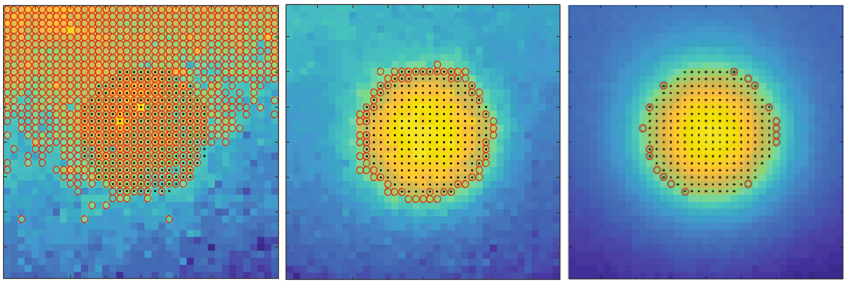


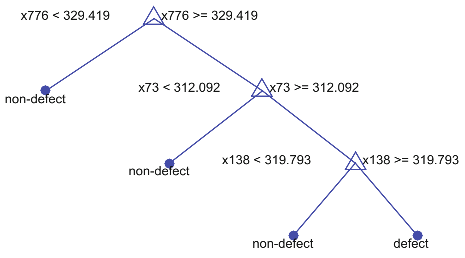
Fig. 9. Thermograms recorded at time instant 50 (left), 100 (middle), and 600 (right). Defect area are marked with black dots. Pixels lying in overlapping region are marked with rings.

i.e. the temperature recorded after the time $\tau = 14.6$ s, and T_{138}, i.e. the temperature recorded after the time $\tau = 27.6$ s. The embedded mechanism of variable selection is a valuable property of decision trees. Note that for classification new instances we do not need to provide entire H-CCs but only three temperature values.

The training set of 1521 instances was split in the root node using T_{776} into left subset containing 1250 instances of "non-defect" class, and right subset containing 268 instances of "defect" class and 3 of "non-defect" class. The right subset is further split using T_{73} into left subset having only two "non-defect" instances, and right subset having 268 "defect" instances and one "non-defect" instance. Finally, the right subset is split using T_{138} giving two leaves with one "non-defect" instance and 268 "defect" instances. Note that all leaves of the tree are pure.

The division of the training set in Fig. 11 is shown. The vertical line represents split-point for T_{776}, which almost perfectly separates two classes. The horizontal line represents split-point for T_{73}, which separates two "non-defect" instances from the rest of the data. Third split using T_{138} (not shown in the figure) separates one "non-defect" instance from 268 "defect" ones.

In the last experiment we test how the reduction in the x-pattern length to $m < n$ first components will affect the classifier accuracy. Now patterns $\mathbf{x}$ include the



```
if $T_{776}$ < 329.419 then
     class = non-defect
else
     if $T_{73}$ < 312.092 then
          class = non-defect
     else
          if $T_{138}$ < 319.793 then
               class = non-defect
          else
               class = defect
```

Fig. 10. Optimal CART and decision rules expressing the tree.

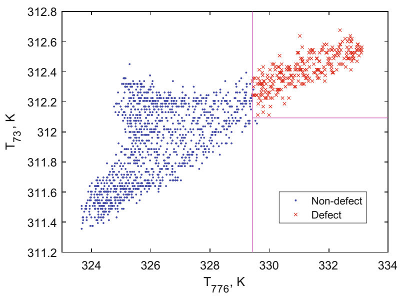


Fig. 11. Training data divided into classes by CART.

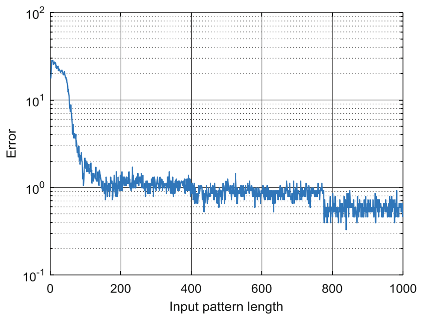


Fig. 12. Classification error depending on the input pattern length.

first parts of H-CCs, from T_1 up to T_m. Figure 12 shows classification error depending on m. When m is too small the temperatures measured on the surface of the sample do not allow the tree to distinguish between classes. For $m > \sim 150$ the error decreases to about 1%, and stays at that level up to $m = 776$. Then the error unexpectedly jumps to the level of about 0.5%, which is rather accidental.

5 Conclusion

In this work, we have demonstrated on the basis of simulation studies that the thermal characteristics of the sample, i.e. the heating-cooling curves, are good information carriers about the subsurface defects in the material. A classification tree used for defect detection gave very low errors, even for short sequences of thermograms including beginning of the heating phase (not less than 30 s). But it should be remembered that our research concerned a simplified case: a fragment of a sample with only one distinct defect, where the area of defect is close to the area of non-defect (balanced case). In real applications, where there are many different defects, which total area is much smaller than the area of the material, higher errors should be expected.

The recursive tree-growing process has embedded mechanism of feature selection, which allowed the tree to select only three points of the heating-cooling curve out of a thousand to classify perfectly all thermogram pixels as “defect” or “non-defect”. This automatic selection of the most informative variables, as well as interpretability (a tree can be expressed as a logical expression which is human understable) and rapid classification are very suitable for material defect detection using active thermography.

References

1. Maldague, X.P.: Theory and Practice of Infrared Technology for Nondestructive Testing. Wiley, Interscience, New York (2001)
2. Minkina, W., Dudzik, S.: Infrared Thermography – Errors and Uncertainties. Wiley, Chichester (2009)
3. Fourier, J.: Théorie du mouvement de la chaleur dans les corps solides – 2 Partie. Mémoires de l’Academie des Sciences **5**, 153 (1826)
4. Ångström, M.A.J.: New method of determining the thermal conductibility of bodies. Phil. Mag. **25**, 130–142 (1863)
5. Dudzik, S.: Estimation of the material defects depth using an active thermography and artificial neural networks, Częstochowa University of Technology Publishers, Częstochowa (2013, in Polish)
6. Dudzik, S.: Application of the naive Bayes classifier to defect characterization using active thermography. J. Nondestr. Eval. **4**(31), 383–392 (2012)
7. Dudzik, S.: Two-stage neural algorithm for defect detection and characterization uses an active thermography. Infrared Phys. Technol. **71**, 187–197 (2015)
8. Dudzik, S.: Analysis of the accuracy of a neural algorithm for defect depth estimation using PCA processing from active thermography data. Infrared Phys. Technol. **56**, 1–7 (2013)
9. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J: Classification and Regression Trees. Wadsworth, Monterey (1984)

Interactive Visualization of Query Results Set from Information Retrieval Using Concept Lattices

Peter Butka(✉), Miroslav Smatana, and Veronika Novotná

Department of Cybernetics and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Letná 9/B, 042 00 Košice, Slovakia
{peter.butka,miroslav.smatana}@tuke.sk,
veronika.novotna@student.tuke.sk

Abstract. This paper is devoted to the interactive visualization of search results obtained by the search engine using the concept lattices. We provide a tool in which the process is realized from the query input, the creation of the concept lattice and then its visualization. The concept lattices are created using Formal Concept Analysis which hierarchically organizes the results in the form of clusters of particular objects composed of the documents with the shared attributes. The resulted concept lattice is able to provide a structured view on the domain of query to the user. This work uses Generalized One-Sided Concept Lattice (GOSCL) for building a hierarchy of concepts. This model is able to create concept lattice from input data tables that contain different types of attributes representing fuzzy sets. Thus, the concept lattice is then shown as an interactive graph on which reductions can be applied, increasing the clarity of the output visualization.

Keywords: Formal concept analysis · One-sided concept lattices · Information retrieval · Reductions · Visualization

1 Introduction

The area of Information Retrieval (IR) deals with providing a structured view of results in order to help understand the searched domain. One of the possibilities is to apply methods from Formal Concept Analysis (FCA) [1] and use them for exploratory data mining. In this case, users are provided with the hierarchical structure of clusters of documents, which represent documents subsets with shared attributes defined within the domain. In standard FCA-based methods work with binary input data tables (so-called crisp case, where an object has/has-not an attribute). However, in practice, it is necessary to process different types of attributes, and for this purpose, several fuzzy approaches were introduced [2–5], or the approach known as Generalized One-Sided Concept Lattice (GOSCL) is a model of one-sided fuzzification (only on one side of data table), where it is possible to process heterogeneous data in the form of the generalized input context with different types of attributes [6].

FCA can be used to solve various problems of IR systems based on questioning and navigation. The results returned by the search engine for the query input are typically formatted as a list of URLs along with the title of the document and its snippet. In the area of information retrieval, the FCA methods were used in systems such as CREDO and its extension CRE-CHAIN-DO [7]. The basic step is to build input context for each query that contains query results as objects and terms found in document title or snippet as attributes. System CREDO is designed only for processing binary contexts that have information about whether the word occurs in the document or not. From the context, a structure consisting of the two-level hierarchy is created. System CRE-CHAIN-DO is specific by creating concepts and joining them into a hierarchy that resembles a tree structure. In this type of hierarchy - tree, the single concept has only one parent with which it shares certain properties. However, this is not a representation of a concept lattice. Hence, FCA is applied only partially. The idea of a concept lattice is to connect concepts which share common attributes. This generally leads to the hierarchy-based structure of domain in which the concepts have more parents and descendants.

Our vision is to apply FCA-based method suitable to work with fuzzy attributes (like GOSCL) and also to provide visualization of the whole concept lattice within the simple IR tool. Therefore, the purpose of this work is to create a structured visualization of search results in the form of concept lattice, where it is possible to use also more complex (fuzzy) attributes. For this reason, GOSCL algorithm is provided for the creation of one-sided fuzzy concept lattices. The result of the work and the main contribution is a visualization tool (browser) which provides such functions and which was implemented and experimentally tested on selected queries.

The paper is organized as follows: Sect. 2 provides some information on FCA, Sect. 3 is related to the reduction of FCA outputs and Sect. 4 describes necessary details regarding GOSCL model. In Sect. 5 the details about the provided tool and its functionalities and implementation are provided. In Sect. 6 we provide simple experiments and discussion on the application of the system. At the end of the paper, we add some conclusions and future work ideas.

2 Formal Concept Analysis (FCA)

When people talk about objects and attributes they formulate their knowledge about the outside world. They express various arguments that some objects share the same values for certain attributes (properties). These relationships between objects and attributes are in most cases represented by a table, where the rows are objects and columns represent attributes. One of the methods of analysis of tabular data is a formal concept analysis. The term FCA is often replaced by term “method of concept lattices”, where the input for analysis of data is tabular data. FCA is a method of exploratory data analysis. This method gives the user non-trivial information about input data, which can be used directly or may be used in the further processing of data.

It has been applied in different areas as knowledge discovery and data/text mining, association rule mining, or information retrieval.

FCA provides two primary outputs:

1. Concept lattice is hierarchically arranged set of clusters - formal concepts that are contained in the input table of data. Formal concepts can be compared with each other. Unless it can be compared, so we can say about them, that one is wider than the other or specific. An important feature of concept lattice is its simple graphical representation. The user can examine the data in graphical form, which incorporates the hierarchical arrangement.
2. Attribute implications describe dependencies between attributes of input data.

There are several available algorithms [8] for the building of concept lattices as NextClosure, UpperNeighbor and Generalized One-Sided Concept Lattice (GOSCL), described in Sect. 4.

3 Concept Lattice Reduction Methods

As we mentioned in the previous section, FCA found its application in many different areas, however, if it is used for analysis of medium or large size data, then generated concept lattice usually contains a large number of concepts and links between them. Such concept lattices are hardly understandable, so some reduction methods are needed to make concept lattice more readable [18]. Basic reduction approaches are based on thresholding (some ranking method is used to evaluate concept quality and concepts with quality lower than the threshold are removed). Another simpler approach can be based on removing concepts which contain less object than some threshold. Also, other approaches were proposed and can be found in [19, 20], or reduction which transform concept lattice into the tree-based structure in [21]. Other possible ways for concept lattice reduction are based on clustering of similar concepts to separate groups and create from them pseudo-concepts or reduce concept lattices based on some entropy-based methods [22].

4 Generalized One-Side Concept Lattice (GOSCL)

For the purpose of this paper, we only describe some basic details regarding the theory of GOSCL [6, 9], some other properties on heterogeneous concept lattices are described in [10, 11]. For generalized one-sided formal context can be considered as a 4-tuple $c = (A, B, L, R)$ that meets the following conditions:

1. B is a non-empty set of all attributes and A is a non-empty set of all objects.
2. $L: B \rightarrow CL$ is a view of a set of attributes to a class all complete of lattices CL. $L(b)$ is the structure of truth values of attribute b for each attribute b.
3. R is a generalized incidence session, $R(a,b) \in L(b)$ for all $b \in B$ a $a \in A$. $R(a,b)$ represents the degree of $L(b)$, which has an object $a \in A$ in attribute b.

A data table for analysis represents the relation R. The ability to create concept lattice from the table that contains attributes of different types is one main difference compared to other approaches.

```
ALGORITHM (Generalized One- Sided Concept Lattice)
1: Input (A, B, L, R) - generalized one- sided concept lattice
2: begin
3:   create lattice  $L := \Pi_{b \in B} L(b)$
4:   $C := \{1_L\}, C \subseteq L$      (set of all intents)
5:   while (A $\neq \emptyset$ )
6:   {
7:     choose a $\in$ A
8:     $C^* := C$
9:     for each c $\in C^*$
10:        $C := C \cup \{c \wedge R\,(a)\}$    (meet operation on lattice)
11:    $A := A\ /\ \{a\}$
12:  }
13:    $C(A,B,L,R) := \emptyset$
14:    for each c $\in$ C
15:      $C(A,B,L,R) := C(A,B,L,R) \cup \{(c',c)\}$
16:  end
17:  Output: $C(A,B,L,R)$    (set of all concepts)
```

This algorithm produces concepts of generalized one-sided concept lattice. It starts with the one concept (defined as 1_L in the algorithm) containing the largest value for every attribute. Operator $\wedge$ is standard lattice operation *meet* on attribute lattices truth value structure defined by *L*. The notation *x*′ is used if we want to find all objects that have at least *x* level of attributes (*x* is subset of attributes values defining the concept). Similarly, we can use same notation for reverse function, e.g., if *y* is some subset of objects defining the concept, then *y*′ is respective definition in attributes values.

5 Visualization of Search Results Using Concept Lattice

In this section, our proposed tool is described. It allows a user to enter a query that is searched by the search engine Google[1]. From the results obtained, an interactive graph of the concept lattice is created. The application creation process consists of 6 main steps from obtaining the results to visualization of the concept lattice.

The first step is **retrieving the results**. The input of the process is the required query and the number of documents which the user chooses in the application settings. This information is processed by GoogleSearch, so it creates the correct request and sends it to the Internet. The request is processed and the response sent to the search engine. The result of the process is specific search results in JSON format.

The second step is **processing of results**. Based on the number of results in the settings, a table with the number of rows corresponding to the number of results is prepared. Thereafter, the documents are processed in a way, where each word from the snippet passes through preprocessing. In this phase, stopwords are deleted, the remaining

[1] https://www.google.com.

words are transformed to lowercase, stemmed and after that, they create a column in the table if the word has not yet existed.

The third step is devoted to the **creation of input context**. The input to this process is a table with information about the count of occurrences of words in documents and the type of context that is represented by the information retrieval model. The application allows the user to choose between a Boolean and a Vector model. To describe the Boolean model, we used a representation of data in the form of a binary value matrix, so 0 if the word in the document is not found, 1 if the word in the document occurs. The Vector model represents a matrix of values of 0 if the word in the document is not found and the frequency of the term in the case that word in the document occurs. Based on the context type selected in settings, the appropriate context is created.

The fourth step is the **creation of a concept lattice and preparation for visualization**. The concept lattice is created through the GOSCL and for visualization of concept lattice, we chose the JUNG[2] project. The graph in JUNG is defined as a set of nodes and a set of edges. Based on this, it is necessary to convert the concept lattice into the appropriate form acceptable by JUNG, where the node will represent the concept and edge relationship between the concepts. We created the node for each concept from the concept lattice into which we inserted this concept. The concept is described by two sets. The first is a set of the object which belongs to the concept and the second set is the ordered set of values that the terms acquire, where the term is determined by the position in the set. Based on this information, we created three ordered sets from the table. The first set contains the actual document objects. The second set contains only those objects of terms that have an occurrence value other than 0 and the third are the occurrence values of these terms. Since the node represents the concept, it was necessary to create a set of oriented edges for each concept of the concept lattice in all its top concepts. Each such edge contains a set of terms that originate from the difference of the term sets from the bottom and top nodes. The resulting nodes and edges are inserted in a graph visualized by JUNG.

The fifth step is a **reduction of concept lattice**. The input to this process is the concept lattice containing all the concepts (nodes) and all the edges between them. We provide two types of reductions that can be selected in the application interface. Based on the settings, the appropriate cropping of original concept lattice is performed and it creates a newly reduced concept lattice. Our application allows the user to trim the concept lattice in two ways. The first reduction method relates to the removal of only the lower, i.e., the most specific node (usually did not contain any document). The second type of reduction uses the support function [17]:

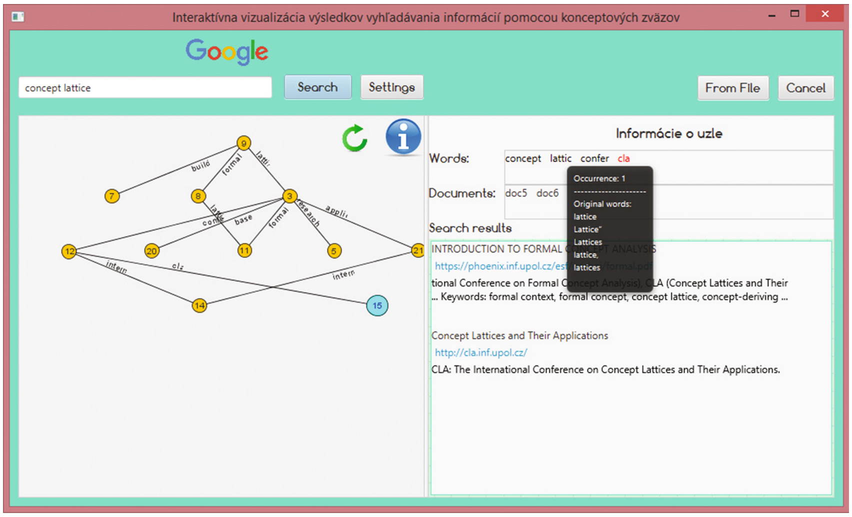
$$support\ (Y) = \frac{|\mathrm{Y}'|}{|\mathrm{A}|} \tag{1}$$

where A is set of all objects and Y is attribute set of some concept for which support is computed. Based on support function we crop a concept lattice based on an input value from a user in the range of <0,1>. It represents the percentage expression of the crop

[2] http://jung.sourceforge.net/.

boundary, which is calculated from the maximum count of documents. Thereafter, we remove all concepts whose number of documents is smaller than the threshold.

The last, sixth step is a **visualization of concept lattice**. The input to the process is a graph created by means of JUNG. Thereafter a visualization is created in which it is defined in terms of the node layout to levels, the appearance of nodes and edges, the interactivity of the graph. The result of the process is an interactive graph, which can be seen in Fig. 1, where we present the user interface of the created application. For node layout, the JUNG library provides a DAG algorithm that is used for directed acyclic graphs. The DAG deploys the nodes through the layers using directed edges in a bottom-up way. Each node with no incoming edges is placed at a zero level. In our case, this node represents the most specific concept. The most generic concept is at the highest level, and it is a node with no outcoming edges. In our application, you can zoom in and zoom out using the mouse wheel within interactivity, using the right mouse button to move the graph. For nodes, we have implemented the tagging, i.e. selecting the node with the left mouse button. By marking the node changes color, size and can be moved within the visualization area. The node representing the concept contains information about words and documents (websites) that are displayed just after clicking on the node.



Words: concept lattic confer cla

Documents: doc5 doc6

Occurrence: 1

Original words:
lattice
Lattice"
Lattices
lattice,
lattices

Search results

INTRODUCTION TO FORMAL C ANALYSIS
https://phoenix.inf.upol.cz/esf formal.pdf
tional Conference on Formal C A (Concept Lattices and Their
... Keywords: formal context, formal concept, concept lattice, concept-deriving ...

Concept Lattices and Their Applications
http://cla.inf.upol.cz/
CLA: The International Conference on Concept Lattices and Their Applications.

Fig. 1. Reduction of fuzzy concept lattice by support for the input query.

As you can see from Fig. 1 our application is divided into several parts:

- The top part is responsible for the input, which will give us the results of the search. However, our system allows not only to process text documents that are currently downloaded from the Internet but also text from the file by button From File in which individual page snippets replace a part of the text contained on the page. The top part is the Settings button that is described below.
- On the left side, there is an interactive visualization area showing a graph of the concept lattice where nodes are organized using DAG layout.

- In the upper right corner of this area, there are picture buttons. The Refresh button, whose task is to redraw the same graph with a different layout of nodes within a level. Part of the area is a blue button showing information about all the search results, which will appear in the right part of the application after clicking on it.
- The right side of the application is responsible for providing two types of information. Information about the documents displayed when the graph is first rendered and information about the node when a node is selected in the graph. Documents information contain keywords, set of words that make difference for current node to its upper (more generic) concept. Within the information provided to a particular result (document), there is a section of words in which all the words appearing in all documents are displayed. Information about the node consists of a list of words and documents representing the concept. Each node can contain old words that node inherits from higher concepts and new words marked in red that occur in the node for the first time. In the Documents section, you can see a list of documents which share common words. Also, a website for each document can be seen in the section Search results.

6 Experiments

In this section, we describe experiments with our proposed application. The measurements within the experiments were repeated for ten different queries. For each measurement, we generated three curves showing the minimum, maximum, and average values for a given measurement. We analyzed the results based on average values.

First experiments were related to the analysis (shown in Fig. 2) of the dependency between the number of search results and the number of created concepts. The measurements took place in the following settings. A minimum number of words in different documents with a value of 1, with no reduction of the concept lattice. As it can be seen, the number of concepts grows almost linearly with the number of documents.

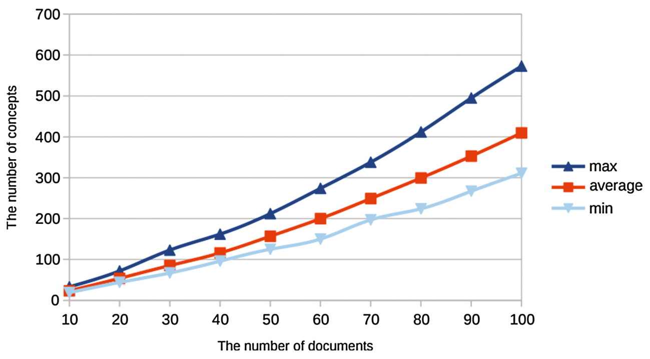


Fig. 2. The dependence of the number of concepts on the number of results for fuzzy concept lattice.

The next experiments (shown in Fig. 3) deals with the effect of support function on the number of concepts of the concept lattice in reduction phase. The measurement shows the percentage of uncut concepts for fuzzy concept lattice. We measured 100 results without reducing input data, where the minimum number of occurrences of words in different documents is set to 1. As you can see, the value of support greatly reduces the count of concepts. Already at the lowest tested value of support (0.1), the number of concepts is reduced to 5%-6% in average. At 0.3, for binary and fuzzy contexts, practically only 1% of concepts remains after the reduction phase.

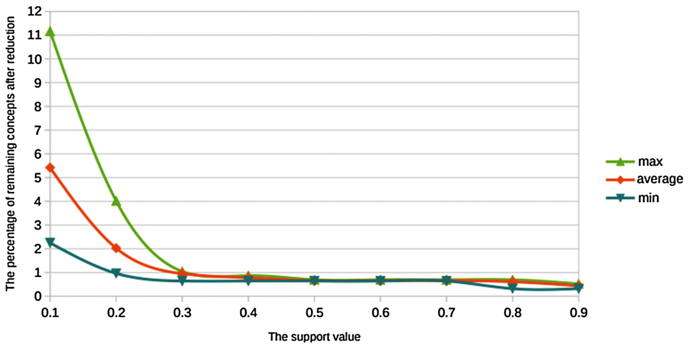


Fig. 3. The dependence of the number of concepts on the support value for fuzzy concept lattice.

In conclusion, we would like to summarize that according to more experiments, the concept lattice without the use of reduction, which is readable to the user, need to be created from a maximum of 10 results. If we use support function, the graph's readability depends on its value. By our experiments, we found that for 100 results, we reached a readable graph for support with a value of 0.2.

From the computation point of view, we have found that most consuming part of the application are actually the step for the search of results through API, and the creation

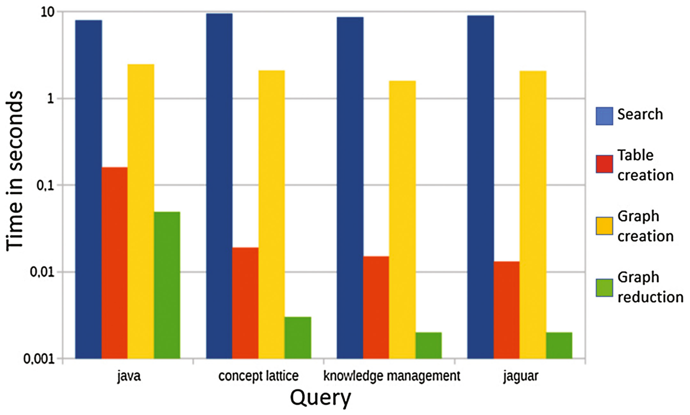


Fig. 4. Application processing time for different queries (in seconds)

of graph (but for a reasonable number of results it is mostly the problem of preparation of JUNG-based visualization, not the computation of lattice). The processing times of particular operations for 100 objects in results set (on different queries) is shown in Fig. 4. For smaller query results sets (under 20–50 objects, depends on the usage of reductions) with a reasonable amount of visualized concepts, the processing time of application is usually under half of one second.

In the future we would like to apply our visualization tool not only in the context of information retrieval or text-mining tasks [12, 13], but also in support of understanding the documents from different domains, like e-learning [14], data on patients from medical domain [15], or other data mining related domains [16].

7 Conclusion

The main aim of this paper was to provide information on the developed tool in Java which is able to process the results obtained from web engine by the selected query and to create concept lattice by one-sided fuzzy concept lattices. The results are hierarchically organized, according to different specific combinations of different types of attributes. The created graph of concept lattice in this tool is enriched with interactive features which allow to create, show and reduce concept lattice based on requests of the user. Based on the implemented solution and experiments we found that individual reductions increase the clarity of the output visualization, improve navigation within the documents. The application in the current state provides space for possible extensions. For example, using fuzzy attributes for other information than just word count in a document. For improving visualization more reduction approaches could be applied. Another possible extension might be to rework the application into a web application using some well-known javascript library for graphs.

Acknowledgments. The work presented in this paper was supported by the Slovak VEGA research grant 1/0493/16, Slovak APVV research grant APVV-16-0213 and Slovak KEGA grant 025TUKE-4/2015.

References

1. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer Verlag, Berlin (1999)
2. Antoni, L., Krajci, S., Kridlo, O., Macek, B., Piskova, L.: On heterogeneous formal contexts. Fuzzy Set. Syst. **234**, 22–33 (2014)
3. Krajci, S.: A generalized concept lattice. Logic J. IGPL **13**(5), 543–550 (2005)
4. Cornejo, M.E., Medina, J., Ramirez-Poussa, E.: Attribute reduction in multi-adjoint concept lattices. Inf. Sci. **294**, 41–56 (2015)
5. Pocs, J.: Note on generating fuzzy concept lattices via Galois connections. Inf. Sci. **185**(1), 128–136 (2012)
6. Butka, P., Pocs, J.: Generalization of one-sided concept lattices. Comput. Inf. **32**(2), 355–370 (2013)

7. Nauer, E., Toussaint, Y.: Dynamical modification of context for an iterative and interactive information retrieval process on the web. In: Concept Lattices and their Applications 2007. CEUR-WS Proceedings (2008)
8. Ganter, B., Obiedkov, S.: An algorithm for closure systems. In: Conceptual Exploration, pp. 39–85. Springer (2016)
9. Butka, P., Pocs, J., Pocsova, J.: On equivalence of conceptual scaling and generalized one-sided concept lattices. Inf. Sci. **259**, 57–70 (2014)
10. Pocs, J., Pocsova, J.: Basic theorem as representation of heterogeneous concept lattices. Front. Comput. Sci. **9**(4), 636–642 (2015)
11. Pocs, J., Pocsova, J.: Bipolarized extension of heterogeneous concept lattices. Appl. Math. Sci. **8**(125–128), 6359–6365 (2014)
12. Sarnovsky, M., Carnoka, N.: Distributed algorithm for text documents clustering based on k-Means approach. Advances in Intelligent Systems and Computing, vol. 430, pp. 165–174 (2016)
13. Sarnovsky, M., Vronc, M.: Distributed boosting algorithm for classification of text documents. In: Proceedings of SAMI 2014 - IEEE 12th International Symposium on Applied Machine Intelligence and Informatics, Herlany, Slovakia, pp. 217–220 (2014)
14. Babic, F., Paralic, J., Bednar, P., Racek, M.: Analytical framework for mirroring and reflection of user activities in E-learning environment. Adv. Intell. Soft Comput. **80**, 287–296 (2010)
15. Babic, F., Majnaric, L., Lukacova, A., Paralic, J., Holzinger, A.: On patients characteristics extraction for metabolic syndrome diagnosis: predictive modelling based on machine learning. Lect. Notes Comput. Sci., vol. 8649, pp. 118–132 (2014)
16. Bartok, J., Babic, F., Bednar, P., Paralic, J., Kovac, J., Bartokova, I., Hluchy, L., Gera, M.: Data Mining for Fog Prediction and Low Clouds Detection. Comput. Inf. **31**(6+), 1441–1464 (2012)
17. Kuznetsov, S.O.: Stability as an estimate of the degree of substantiation of hypotheses derived on the basis of operational similarity. In: Automatic Documentation and Mathematical Linguistics (1990)
18. Dias, S.M., Vieira, N.J.: Concept lattices reduction: definition, analysis and classification. Expert Syst. Appl. **42**(20), 7084–7097 (2015)
19. Butka, P., Pocs, J., Pocsová, J.: Reduction of concepts from generalized one-sided concept lattice based on subsets quality measure. Advances in Intelligent Systems and Computing, vol. 314, pp. 101–111 (2015)
20. Antoni, L., Krajci, S., Kridlo, O.: Randomized fuzzy formal contexts and relevance of one-sided concepts. LNAI (Subseries of LNCS), vol. 9113, pp. 183–199 (2015)
21. Melo, C., Le-Grand, B., Aufaure, A.: Browsing large concept lattices through tree extraction and reduction methods. Int. J. Intell. Inf. Technol. (IJIIT) **9**(4), 16–34 (2013)
22. Singh, P.K., Cherukuri, A.K., Li, J.: Concepts reduction in formal concept analysis with fuzzy setting using Shannon entropy. Int. J. Mach. Learn. Cybern. **8**(1), 179–189 (2017)

PID-Fuzzy Controllers with Dynamic Structure and Evolutionary Method for Their Construction

Krystian Łapa(✉) and Krzysztof Cpałka

Institute of Computational Intelligence,
Częstochowa University of Technology, Częstochowa, Poland
{krystian.lapa,krzysztof.cpalka}@iisi.pcz.pl

Abstract. The controllers are interesting type of information systems. Their structure and parameters for atypical applications usually are selected by trial and error method, which can be time-consuming. In this paper a controller structure, which is an ensemble of PID controller and fuzzy system, and an automatic evolutionary method for its construction is presented. The significant feature of proposed method is that the controller elements, that increase its complexity but do not improve the controller precision in the sense of the adopted evaluation function, can be reduced. Moreover, this method allows to use the knowledge of the controlled object. A typical control problem has been used to test the authors approach.

Keywords: Evolutionary method · Controller · PID · FIR · Fuzzy system

1 Introduction

The controllers are interesting type of information systems. The controllers goal is to affect an controlled object to obtain expected results (for example a specific state of the object). The most often used in practice are PID controllers (see e.g. [1, 8]). These controllers are based on three elements (three-term controllers): proportional (P), integral (I) and differential (D). Using of P, I and D elements allows for processing respectively: current error, accumulation of deviations from the past and predicted future deviations. In the literature many controllers that use combination of P, I and D elements can be found: PI with feed-forward [10], PID with additional low-pass [11], PID with anti-windup and compensation mechanism [15], pseudo-derivative feedback with feed-forward gain (PDFF) [4], PID controllers that process multiple input signals [3, 13], etc. Among other types of controllers fuzzy systems based controllers (see e.g. [16]) have a significant meaning. The fuzzy systems belong to computational intelligence methods (see e.g. [2]). Their advantage is that their fuzzy rules contain interpretable information about how to generate a control signal. However they cannot process accumulation of deviations from the past and predicted future deviations.

In the process of controller construction the following properties are usually taken into consideration: (a) rise time, (b) overshoot, (c) settling time, (d) steady-state error, and (e) stability. Due to this reason it is important to use the filters on controller input signals [17]. Such signals are often burdened by noise or digital resolution of the sensors used to measure them. One of the most commonly used filters are finite impulse response filters (FIR, [1]). The feature of such a filter is that when its input is given as a signal with finite length (in a time domain) the output value has also finite length. This is due to the fact that the FIR filters (as opposed to infinite impulse response-IIR) have no feedbacks. All this makes the design of the controllers for special applications (for which there is no clear methodology in the literature and which is usually solved by trial and error method) a complex issue [7].

In our previous work we discussed the controllers, which structure is an ensemble of PID controller and fuzzy system with FIR filters (FIR+PID+FS, [9]). However, such regulators use a full set of P, I, D processing elements for each input signal, and a full structure of the fuzzy system (resulting from, for example, the set of input parameters). The approach introduced in this paper enables the automatic reduction of the proposed structure, which is a novel feature. This was achieved by using: (a) flexible (dynamic) structure of the fuzzy system, and (b) dedicated evolutionary learning algorithm. This solution has not been discussed before in literature. This is important because: (a) it simplifies hardware implementation, (b) simplifies fuzzy rules writing, and (c) it makes easier to interpret how the controller works.

The structure of this work is as follows: in Sect. 2 proposed controller is described, in Sect. 3 an evolutionary method for controller construction is provided, in Sect. 4 a simulation results are presented, and in Sect. 5 a conclusions are drawn.

2 Description of Proposed Controller

The structure of proposed controller is presented in Fig. 1. It consist of four layers that are described later in this section.

2.1 Filtration Layer

Finite impulse response filters (FIR) are one of the most commonly used filters. The purpose of FIR filters is to smoothen the signal values. It is achieved by averaging weighted values from consecutive time steps. The output value of the filter depends on the filter length and its numerical parameters. Although FIR filters are used to improve the quality of control and to reduce impact of signal noise, they might decrease the controller efficiency [17]. This is due to the fact that the filtration adds a side effect - phase delay of the filtered signal.

The filtration layer of the controller (see Fig. 1) contains a FIR filters. For each controller input signal output values of filtration layer are calculated as follows:

$$e_q^{\mathrm{FIL}}(k) = \sum_{l=0}^{s_q^{\mathrm{FIL}}-1} b_{q,l}^{\mathrm{FIL}} \cdot e_q^{\mathrm{SIG}}(k-l), \tag{1}$$

where s_q^{FIL} stands for odd length of the filter (filter has to have a middle element), $q = 1, \ldots, Q$ stands for input signal index, Q stands for number of input signals,

$e_q^{\mathrm{SIG}}(k-l)$ stands for input signal from $k-l$ time step, $b_{q,l}^{\mathrm{FIL}}$ stands for the weight of the signal for $k-l$ time step calculated as follows:

$$b_{q,l}^{\mathrm{FIL}} = \begin{cases} \frac{\sin\left(2\cdot\pi\cdot ft_q^{\mathrm{FIL}}\cdot\left(l-0.5\cdot\left(s_q^{\mathrm{FIL}}-1\right)\right)\right)}{\pi\cdot\left(l-0.5\cdot\left(s_q^{\mathrm{FIL}}-1\right)\right)} & \text{for} \quad l \neq 0.5\cdot\left(s_q^{\mathrm{FIL}}-1\right) \\ 2\cdot ft_q^{\mathrm{FIL}} & \text{for} \quad l = 0.5\cdot\left(s_q^{\mathrm{FIL}}-1\right), \end{cases} \tag{2}$$

where ft_q^{FIL} stands for frequency of the filters. In this paper values of parameters s_q^{FIL} and ft_q^{FIL} are selected automatically.

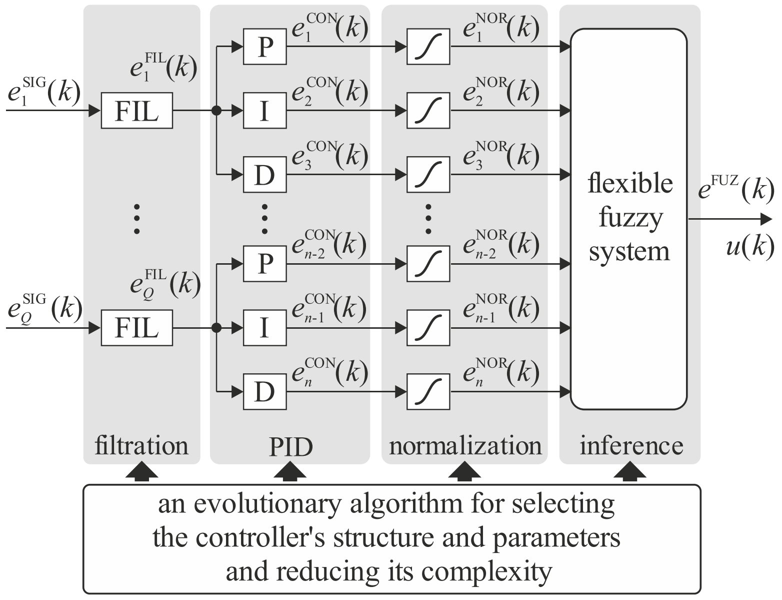


Fig. 1. Proposed controller structure.

2.2 PID Layer

The PID layer of the controller (see Fig. 1) contains three parallel placed elements (P, I and D). Outputs of this layer are calculated as follows:

$$\begin{cases} e_{3q-2}^{\mathrm{CON}}(k) = K_{3q-2}^{\mathrm{CON}} \cdot e_q^{\mathrm{FIL}}(k) \\ e_{3q-1}^{\mathrm{CON}}(k) = K_{3q-1}^{\mathrm{CON}} \cdot \sum\limits_{k2=0}^{k2=k} e_q^{\mathrm{FIL}}(k2) \\ e_{3q}^{\mathrm{CON}}(k) = K_{3q}^{\mathrm{CON}} \cdot \left(e_q^{\mathrm{FIL}}(k) - e_q^{\mathrm{FIL}}(k-1)\right), \end{cases} \tag{3}$$

where K_{3q-2}^{CON} is enforcement parameter of element P, $K_{3q-1}^{\mathrm{CON}} = T_s/T^{\mathrm{I}}$ (T^{I} stands for integral time constant, and T_s stands for control time step), $K_{3q}^{\mathrm{CON}} = T^{\mathrm{D}}/T_s$ (T^{D} stands for differential time constant) ($i = 1, \ldots, n$), and n stands for the number of parameters multiplied by number of input signals (each filtered signal generate three outputs $n = 3 \cdot Q$). The parameters $K_1^{\mathrm{CON}}, \ldots, K_n^{\mathrm{CON}}$ are selected automatically by proposed evolutionary method.

2.3 Normalization Layer

The purpose of normalization layer of the introduced controller (see Fig. 1) is to adjust (normalize) the signal values to fit the role of fuzzy system inputs. This normalization is executed as follows:

$$e_i^{\mathrm{NOR}}(k) = p3_{d_2(i)}^{\mathrm{NOR}} \cdot \frac{1}{1 + \exp\left(-\left(p1_{d_2(i)}^{\mathrm{NOR}} \cdot e_i^{\mathrm{CON}}(k) - p2_{d_2(i)}^{\mathrm{NOR}}\right)\right)} + p4_{d_2(i)}^{\mathrm{NOR}}, \tag{4}$$

where $p1_q^{\mathrm{NOR}}$, $p2_q^{\mathrm{NOR}}$, $p3_q^{\mathrm{NOR}}$, $p4_q^{\mathrm{NOR}}$ stand for parameters of normalization function, calculated as follows:

$$\begin{cases} p1_q^{\mathrm{NOR}} = \dfrac{-\log\left(-\frac{y^{\mathrm{CST}}}{y^{\mathrm{CST}} - e_q^{\mathrm{MAX}} + e_q^{\mathrm{MIN}}}\right) + \log\left(-\frac{y^{\mathrm{CST}} - e_q^{\mathrm{MAX}} + e_q^{\mathrm{MIN}}}{y^{\mathrm{CST}}}\right)}{e_q^{\mathrm{MAX}} - e_q^{\mathrm{MIN}}} \\ p2_q^{\mathrm{NOR}} = \dfrac{-e_q^{\mathrm{MIN}} \cdot \log\left(-\frac{y^{\mathrm{CST}}}{y^{\mathrm{CST}} - e_q^{\mathrm{MAX}} + e_q^{\mathrm{MIN}}}\right) + e_q^{\mathrm{MAX}} \cdot \log\left(-\frac{y^{\mathrm{CST}} - e_q^{\mathrm{MAX}} + e_q^{\mathrm{MIN}}}{y^{\mathrm{CST}}}\right)}{e_q^{\mathrm{MAX}} - e_q^{\mathrm{MIN}}} \\ p3_q^{\mathrm{NOR}} = y^{\mathrm{MAX}} - y^{\mathrm{MIN}} \\ p4_q^{\mathrm{NOR}} = y^{\mathrm{MIN}}, \end{cases} \tag{5}$$

where e_q^{MIN} and e_q^{MAX} stand for minimum and maximum of acceptable values of the signals $e_q^{\mathrm{SIG}}(k)$, y^{CST} stands for small value resulting from an infinite medium of sigmoid function, y^{MIN} and y^{MAX} stand for lower and upper values of normalization function.

2.4 Inference Layer

The inference layer of the controller (see Fig. 1) is based on a fuzzy system of the Mamdani type. It works on the basis of fuzzy rules. The fuzzy rules allow for interpretation of system behavior which is a significant advantage of such systems. In case of considered in this paper Mamdani system with singleton fuzzification [16], the fuzzy rules notation can be defined as follows (according to the controller notation used in this paper):

$$\mathrm{R}^b : \left(\text{IF } e_1(k) \text{ is } A_1^b \text{ AND } \ldots \text{ AND } e_n(k) \text{ is } A_n^b \text{ THEN } u(k) \text{ is } B^b\right), \tag{6}$$

where b stands for fuzzy rule index ($b = 1, \ldots, N$), $e_i(k)$ stands for controller input signals ($i = 1, \ldots, n$), $u(k)$ stands for controller output signal, A_i^b stands for input fuzzy sets determined by the membership functions $\mu_{A_i^b}(\bar{x}_i)$ ($i = 1, \ldots, n$), B^b stands for output fuzzy sets determined by the membership functions $\mu_{B^b}(\bar{y})$.

Output signal $\bar{u}(k)$ of the fuzzy system can be calculated with center of area method (details can be found in our previous works, see e.g. [5, 8, 16]):

$$\bar{u}(k) = \frac{\sum\limits_{r=1}^{N} \bar{y}_r \cdot \mathop{S}\limits_{b=1}^{N} \left\{ T\left\{ \tau_b(\mathbf{e}^{\mathrm{NOR}}(k)), \mu_{B^b}\left(\overline{yB}_r^{\mathrm{SYS}}\right) \right\} \right\}}{\sum\limits_{r=1}^{N} \mathop{S}\limits_{b=1}^{N} \left\{ T\left\{ \tau_b(\mathbf{e}^{\mathrm{NOR}}(k)), \mu_{B^b}\left(\overline{yB}_r^{\mathrm{SYS}}\right) \right\} \right\}}, \tag{7}$$

where $\overline{yB}_r$ stands for centers of output fuzzy sets ($r = 1, \ldots, N$), $S\{\cdot\}$ and $T\{\cdot\}$ stand for triangular norms (t-norms and t-conorms, [6]). The firing level of the each fuzzy rule (6) of system (7) is calculated as follows:

$$\tau_b\left(\mathbf{e}^{\mathrm{NOR}}(k)\right) = T^* \left\{ \begin{array}{c} \mu_{A_1^b}\left(e_1^{\mathrm{NOR}}(k)\right), \ldots, \mu_{A_n^b}\left(e_n^{\mathrm{NOR}}(k)\right); \\ C_1^{\mathrm{INP}} \cdot C_{1,b}^{\mathrm{SET}}, \ldots, C_n^{\mathrm{INP}} \cdot C_{n,b}^{\mathrm{SET}} \end{array} \right\}, \tag{8}$$

where $C_i^{\mathrm{INP}} \in \{0, 1\}$ and $C_{i,b}^{\mathrm{SET}} \in \{0, 1\}$ are binary "keys" that determine whether respectively fuzzy system input or fuzzy set is active or reduced from the system (element is active if the value of key equals 1, element is reduced if the value of the key equals 0), and $T^*\{.\}$ is the weighted t-norm [16] in the form:

$$\begin{array}{l} T^* \left\{ \begin{array}{c} a_1, a_2; \\ w_1, w_2 \end{array} \right\} = T \left\{ \begin{array}{c} 1 - w_1 \cdot (1 - a_1), \\ 1 - w_2 \cdot (1 - a_2) \end{array} \right\} \\ \overset{\text{e.g.}}{=} (1 - w_1 \cdot (1 - a_1)) \cdot (1 - w_2 \cdot (1 - a_2)). \end{array} \tag{9}$$

Values w_1 and $w_2 \in [0, 1]$ in the formula (9) mean weights of importance of the arguments $a_1, a_2 \in [0, 1]$. Please note that $T^*\{a_1, a_2; 1, 1\} = T\{a_1, a_2\}$ and $T^*\{a_1, a_2; 1, 0\} = a_1$.

3 Description of Evolutionary Method for Controller Construction

Method of construction of the proposed controller is based on knowledge about modelled object (process), which allows to automate controller selection phase without risk of damaging of real object. Moreover, proposed method is on a genetic algorithm. This algorithm belongs to computational intelligence methods that are used mostly to solve optimization problems. Their main characteristic is that they are capable of finding approximate solutions in acceptable time. Genetic algorithms are part of population-based algorithms [16], where process of finding solution is based on modification (processing) of the group of individuals (population). Each individual encodes a complete solution to the problem under consideration.

3.1 Encoding of the Individuals

The proposed encoding of the individuals is based on Pittsburgh approach, where single individual $\mathbf{X}_{ch}$ encodes information about controller. Presented controller consist of numerical parameters of its elements and binary parameters that allow to reduce its structure. Thus, each individual takes the following form:

$$\mathbf{X}_{ch} = \left\{\mathbf{X}_{ch}^{\mathrm{PAR}}, \mathbf{X}_{ch}^{\mathrm{STR}}\right\}, \tag{10}$$

where $\mathbf{X}_{ch}^{\mathrm{PAR}}$ contains numerical parameters defined as follows:

$$\mathbf{X}_{ch}^{\mathrm{PAR}} = \left\{ \begin{array}{c} s_1^{\mathrm{FIL}}, \ldots, s_Q^{\mathrm{FIL}}, ft_1^{\mathrm{FIL}}, \ldots, ft_Q^{\mathrm{FIL}}, K_1^{\mathrm{CON}}, \ldots, K_n^{\mathrm{CON}}, \\ \overline{xA}_{1,1}^{\mathrm{SYS}}, \ldots, \overline{xA}_{n,1}^{\mathrm{SYS}}, \ldots, \overline{xA}_{1,N}^{\mathrm{SYS}}, \ldots, \overline{xA}_{n,N}^{\mathrm{SYS}}, \\ \sigma A_{1,1}^{\mathrm{SYS}}, \ldots, \sigma A_{n,1}^{\mathrm{SYS}}, \ldots, \sigma A_{1,N}^{\mathrm{SYS}}, \ldots, \sigma A_{n,N}^{\mathrm{SYS}}, \\ \overline{yB}_1^{\mathrm{SYS}}, \ldots, \overline{yB}_N^{\mathrm{SYS}}, \sigma B_1^{\mathrm{SYS}}, \ldots, \sigma B_N^{\mathrm{SYS}} \end{array} \right\}, \tag{11}$$

where $K_1^{\mathrm{CON}}, \ldots, K_n^{\mathrm{CON}}$ represents all parameters from Eq. (3), $\overline{xA}_{i,b}^{\mathrm{SYS}}$ and $\sigma A_{i,b}^{\mathrm{SYS}}$ are centers and sizes of chosen in this work Gaussian membership functions that represents input fuzzy sets A_i^b, $\overline{yB}_b^{\mathrm{SYS}}$ and $\sigma B_b^{\mathrm{SYS}}$ are centers and widths of chosen in this work Gaussian membership functions that represents output fuzzy sets B^b. The binary parameters $\mathbf{X}_{ch}^{\mathrm{STR}}$ are defined as follows:

$$\mathbf{X}_{ch}^{\mathrm{STR}} = \left\{ C_1^{\mathrm{INP}}, \ldots, C_n^{\mathrm{INP}}, C_{1,1}^{\mathrm{SET}}, \ldots, C_{n,1}^{\mathrm{SET}}, \ldots, C_{1,N}^{\mathrm{SET}}, \ldots, C_{n,N}^{\mathrm{SET}} \right\}. \tag{12}$$

3.2 Processing of the Individuals

Encoding of the individuals (10) enables using the population-based algorithms to find a satisfactory set of parameters. Typical population-based algorithms are designed to process exclusively the numerical or binary values. Therefore, mechanisms from different algorithms can be combined to process both types of values [5]. Used in this paper optimization method is based on genetic algorithm. In the first step of the algorithm a population of N^{init} individuals is generated randomly. Next, these individuals are evaluated by fitness function that assigns them values that defines their adaptation (quality) in the population. In the second step offspring population is generated. For this purpose parent individuals are selected from the base population (by any type of selection mechanism, see e.g. [16]) and on their basis the offspring individuals are formed. To create them crossover and mutation genetic operators are used to modify numerical values and binary genetic mutation operator is used to modify binary values. The offspring individuals are evaluated by fitness function as well. Next, the individuals for the next generation (iteration) of the algorithm are selected. The common approach is the selection of best N^{POP} individuals (according to fitness function values) from both parent and offspring population. In the last step of the algorithm a stop condition is checked. If this condition is met (for example specified number of iterations was achieved) the algorithm stops and the best individual is presented. In the other case the algorithm goes back to the second step. More details can be found in e.g. [5, 12, 16].

4 Simulations

The simulations were aimed at showing the possibilities of reducing controller elements and on increasing controller efficiency with active filters of the signals. Therefore, the simulation variants shown in the Table 1 have been tested.

Table 1. Considered simulation cases ('*' denoted a number of fuzzy rules).

Case name	Possible reduction of inputs (C^{INP})	Possible reduction of fuzzy sets (C^{SET})	Binary parameters limitations
*NN	No	No	$C_i^{\mathrm{INP}} = 1$ and $C_{i,b}^{\mathrm{SET}} = 1$
*NY	No	Yes	$C_i^{\mathrm{INP}} = 1$ and $C_{i,b}^{\mathrm{SET}} \in \{0,1\}$
*YN	Yes	No	$C_i^{\mathrm{INP}} \in \{0,1\}$ and $C_{i,b}^{\mathrm{SET}} = 1$
*YY	Yes	Yes	$C_i^{\mathrm{INP}} \in \{0,1\}$ and $C_{i,b}^{\mathrm{SET}} \in \{0,1\}$

In the simulations the Mass-Spring-Dump problem was used [8] with the following criteria used for evaluation: (a) RMSE between expected state of the object and actual state, (b) oscillations of the controller output (OSC - see [8]), (c) overshooting of the actual object state (OVH - see [8]), and (d) number of active fuzzy sets (SET). These criteria were integrated in the fitness function (FF) used to evaluate the individuals (encoded according to Eq. (10)):

$$\mathrm{ff}(\mathbf{X}_{ch}) = w_1 \cdot \mathrm{RMSE} + w_2 \cdot \mathrm{OSC} + w_3 \cdot \mathrm{OVH} + w_4 \cdot \mathrm{SET}, \tag{13}$$

where $w_1 = 5.00$, $w_2 = 0.20$, $w_3 = 1.00$, and $w_4 = 0.02$ stands for chosen weights of fitness function components. According to Eq. (13) the smaller values of the fitness function mean better performance of the individual. In case of greater number of components of the evaluation function, formula (13) should be replaced by another method of multi-criterion optimization.

The simulation goal is to put position s^1 of mass m^1 into expected position s^* (the s^* changes in time) with smallest error of fitness function value as possible. More details about simulation problem can be found in [8].

For the simulations the parameters of controller (see Fig. 1) were set as follows: $Q = 3$ ($e_1^{\mathrm{SIG}}(k) = s^1 - s^*$, $e_2^{\mathrm{SIG}}(k) = s^1$, $e_3^{\mathrm{SIG}}(k) = s^*$), $T_s = 0.0001$ s, type of triangular norms: algebraic, the noise of the input signals: 1%.

For the optimization algorithm (described in Sect. 3, the following parameters were set: number of individuals: 100, number of iterations: 1000, crossover probability: 0.90, mutation probability: 0.30, mutation range: 0.15, binary mutation probability: 0.10, selection method: roulette wheel, number of simulations for each case: 50 (the results were averaged). The minims and maxims of parameters from Eq. (11) were set

Table 2. Averaged simulation results (BST stands for best obtained RMSE, and PID stands for average number of active P, I, and D elements). Significant results were marked in bold.

Controller/Case	Filters	N	C^{INP}	C^{SET}	FF	RMSE	BST	OSC	OVH	PID	SET
PID+FS [8]	No	3	Yes	No	1.625	**0.128**	0.090	2.450	0.195	4.5	13.5
Cascade PID [14]	Yes	3	No	No	3.663	0.144	**0.076**	13.767	0.190	7.5	–
3NN (this paper)	Yes	3	No	No	2.858	0.191	0.091	3.644	0.575	9.0	27.0
3NY (this paper)	Yes	3	No	yes	1.603	0.170	0.088	2.081	**0.105**	9.0	10.5
3YN (this paper)	Yes	3	Yes	no	**1.431**	0.172	0.112	**1.483**	0.170	**1.6**	**4.8**
3YY (this paper)	Yes	3	Yes	Yes	1.532	0.197	0.129	**1.647**	0.142	**1.5**	**3.5**
5NN (this paper)	Yes	5	No	No	2.339	**0.126**	**0.065**	2.281	0.255	9.0	45.0
5NY (this paper)	Yes	5	No	Yes	1.702	**0.127**	**0.084**	2.548	**0.137**	9.0	18.8
5YN (this paper)	Yes	5	Yes	No	**1.456**	0.156	0.108	**1.412**	0.190	1.8	9.1
5YY (this paper)	Yes	5	Yes	Yes	**1.501**	0.165	0.128	1.715	**0.137**	**1.7**	**8.8**

as follows: $s_q^{\mathrm{FIL}} \in [5, 25]$, $ft_q^{\mathrm{FIL}} \in [0.1, 0.5]$, $K_i^{\mathrm{CON}} \in [-1, 1]$, $\overline{xA}_{i,b}^{\mathrm{SYS}} \in [-1, 1]$, $\overline{yB}_b^{\mathrm{SYS}} \in [-1, 1]$, $\sigma A_{i,b}^{\mathrm{SYS}} \in [0.1, 0.3]$, and $\sigma B_b^{\mathrm{SYS}} \in [0.1, 0.3]$.

The obtained results are presented in Table 2. The fuzzy rules notation and fuzzy rules visualization for best controllers (according to fitness function value) are shown in Fig. 3 and in Table 3. The simulations visualization of signals s^1, s^* and output of the controller $u(k)$ for examples of 3YN and 5YN controllers are presented in Fig. 2.

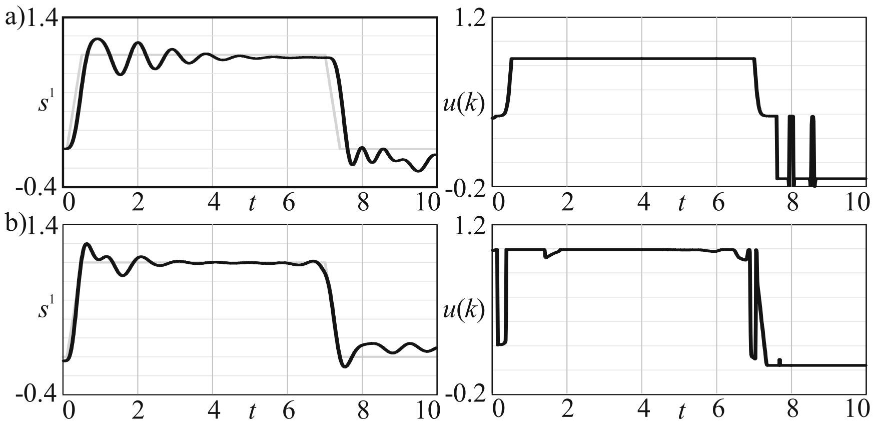


Fig. 2. The simulations visualization of signals s^1, s^* and $u(k)$ for examples of: (a) 3YN and (b) 5YN controllers. The grey line stands for expected state of the object (signal s^*).

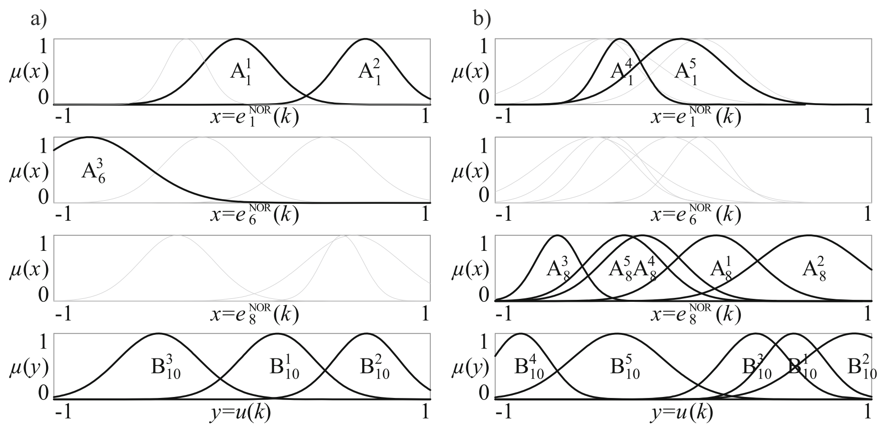


Fig. 3. Fuzzy rules notation of best obtained controllers for: (a) $N = 3$, (b) $N = 5$. The fuzzy sets for reduced inputs were skipped ($e_2^{NOR}(k)$ $-e_5^{NOR}(k)$, $e_7^{NOR}(k)$, $e_9^{NOR}(k)$). At the same time the reduced fuzzy sets were marked grey.

Simulation conclusions can be summed up as follows:

- The controllers based on fuzzy system (7) generated output signal with small amplitude of oscillation (Table 2, column OSC).
- The controllers based on a fuzzy system with possible reduction of its elements had smaller overshooting in comparison to controllers with static structure (Table 2, column OVH) and lower complexity (Table 2, columns PID and SET).
- Using FIR filters slightly weakened controllers efficiency in the sense of fitness function (13) (Table 2, column FF, PID-FUZZY vs. 3NN/5NN). However, the use of controllers with possible reduction of its components eliminated this weakness.
- The best RMSE values were obtained by controllers with possible reduction of fuzzy sets (Table 2, column RMSE, 3NY/5NY). Simultaneously controllers with possible reduction of fuzzy sets and inputs were characterized by simplest structures and satisfying RMSE (Table 2, columns: RMSE/PID/SET, 3YY/5YY)
- The controllers based on fuzzy system with higher number of fuzzy rules ($N = 5$) and possible reduction of fuzzy sets perform best in sense of fitness function (13). At the same time, for 5NN and 5NY cases, obtained RMSE was better than in case of controllers without using filters.
- The fuzzy rules presented in Table 3 and in Fig. 3 are simple and interpretable. At the same time, the best obtained cases of controllers quickly adjust the position of the mass (s^1) to the expected position (s^*) and their performance is satisfactory in the sense of the adopted fitness function (13) (see Fig. 2).

Table 3. Fuzzy rules notation of the fuzzy sets presented in Fig. 3.

N	Fuzzy rules notation
3	$\begin{cases} R^1 : \left(\text{IF } e_1^{\text{NOR}}(k) \text{ is } A_1^1 \text{ THEN } u(k) \text{ is } B^1\right) \\ R^2 : \left(\text{IF } e_1^{\text{NOR}}(k) \text{ is } A_1^2 \text{ THEN } u(k) \text{ is } B^2\right) \\ R^3 : \left(\text{IF } e_6^{\text{NOR}}(k) \text{ is } A_6^3 \text{ THEN } u(k) \text{ is } B^3\right) \end{cases}$
5	$\begin{cases} R^1 : \left(\text{IF } e_8^{\text{NOR}}(k) \text{ is } A_8^1 \text{ THEN } u(k) \text{ is } B^1\right) \\ R^2 : \left(\text{IF } e_8^{\text{NOR}}(k) \text{ is } A_8^2 \text{ THEN } u(k) \text{ is } B^2\right) \\ R^3 : \left(\text{IF } e_8^{\text{NOR}}(k) \text{ is } A_8^3 \text{ THEN } u(k) \text{ is } B^3\right) \\ R^4 : \left(\text{IF } e_1^{\text{NOR}}(k) \text{ is } A_1^4 \text{ AND } e_8^{\text{NOR}}(k) \text{ is } A_8^4 \text{ THEN } u(k) \text{ is } B^4\right) \\ R^5 : \left(\text{IF } e_1^{\text{NOR}}(k) \text{ is } A_1^5 \text{ AND } e_8^{\text{NOR}}(k) \text{ is } A_8^5 \text{ THEN } u(k) \text{ is } B^5\right) \end{cases}$

5 Conclusions

The PID-fuzzy controllers with dynamic structure and evolutionary method for its construction proposed in this work performed very well in the simulations. It allowed to significantly reduce the complexity of the controllers while maintaining their acceptable operating accuracy, overshooting and oscillation. The use of the reduction also improved the performance of fuzzy controllers with FIR filters (in the sense of accepted evaluation criteria) in comparison to the controllers without possible reduction of their structures.

The evolutionary construction of the controllers with reducible structures can be applied, among the others, for a system where the controller should perform accurate and should be characterized by simple structure (due, for example, to the requirement of hardware implementation).

Acknowledgment. The project was financed by the National Science Centre (Poland) on the basis of the decision number DEC-2012/05/B/ST7/02138.

The authors would like to thank the reviewers for very helpful suggestions and comments in the revision process.

References

1. Alia, M.A.K., Younes, T.M., Alsabbah, S.A.: A design of a PID self-tuning controller using LabVIEW. J. Softw. Eng. Appl. **4**, 161–171 (2011)
2. Borzemski, L., Kliber, M., Nowak, Z.: Using data mining algorithms in web performance prediction. Cybern. Syst. Int. J. **40**(2), 176–187 (2009)
3. Boyd, S., Hast, M., Åström, K.J.: MIMO PID tuning via iterated LMI restriction. Int. J. Robust Nonlinear Control **26**, 1718–1731 (2016)
4. Cheng, S., Li, C.W.: Fuzzy PDFF-IIR controller for PMSM drive systems. Control Eng. Pract. **19**, 828–835 (2011)
5. Cpałka, K.: Design of Interpretable Fuzzy Systems. Springer, Heidelberg (2017)
6. Klement, E.P., Mesiar, R., Pap, E.: Triangular Norms. Springer, New York (2000)
7. Lan, K., Sekiyama, K.: Autonomous viewpoint selection of robot based on aesthetic evaluation of a scene. J. Artif. Intell. Soft Comput. Res. **6**(4), 255–265 (2016)

8. Łapa, K., Szczypta, J., Saito, T.: Aspects of evolutionary construction of new flexible PID-fuzzy controller. Artif. Intell. Soft Comput. **9692**, 450–464 (2016)
9. Łapa, K., Cpałka, K., Przybył, A., Saito, T.: Fuzzy PID controllers with FIR filtering and a method for their construction. In: Artificial Intelligence and Soft Computing. Springer (2017). (in print)
10. Leva, A., Papadopoulos, A.V.: Tuning of event-based industrial controllers with simple stability guarantees. J. Process Control **23**, 1251–1260 (2013)
11. Maggio, M., Bonvini, M., Leva, A.: The PID + p controller structure and its contextual autotuning. J. Process Control **22**, 1237–1245 (2012)
12. Melanie, M.: An Introduction to Genetic Algorithms. MIT Press, Cambridge (1999)
13. Pamar, K., Arvapalli, R., Sadhu, Y., Viswaraju, S.: Cascaded PID controller design for heating furnace temperature control. IOSR J. Electron. Commun. Eng. **5**(3), 76–83 (2013)
14. Przybył, A., Łapa, K., Szczypta, J., Wang, L.: The method of the evolutionary designing the elastic controller structure. In: Artificial Intelligence and Soft Computing, vol. 9692, pp. 476–492 (2016)
15. Ribica, A.I., Mataušek, M.R.: A dead-time compensating PID controller structure and robust tuning. J. Process Control **22**, 1340–1349 (2012)
16. Rutkowski, L.: Computational Intelligence. Springer, Heidelberg (2008)
17. Segovia, R.V., Hägglund, T., Aström, K.J.: Noise filtering in PI and PID control. In: American Control Conference, pp. 1763–1770 (2013)

Community Detection in Bibsonomy Using Data Clustering

Zakaria Saoud[1(✉)] and Jan Platoš[2]

[1] Computer Sciences Department,
University of Sciences and Technologies Houari Boumediene,
BP 32 EL ALIA, 16111 Bab Ezzouar, Algiers, Algeria
zakaria.saoud@live.fr

[2] VŠB - Technical University of Ostrava,
17. Listopadu 15, Ostrava, Czech Republic
jan.platos@vsb.cz

Abstract. Community detection aims to extract the related groups of nodes from complex networks, by exploiting the network topology. Different approaches have been proposed for community detection, where most of them are based on clustering algorithms. In this paper we investigate how we can use the clustering for the community detection in the academic social bookmarking website: Bibsonomy. Our goal is to determine the most suitable clustering algorithm for similar user detection in Bibsonomy. To realize that, we have compared three clustering algorithms: The k-means, the k-medoids and the Agglomerative clustering algorithms. Experimental results demonstrate that k-means performs better than the other algorithms, for community detection in Bibsonomy.

Keywords: Community detection · Data visualization · Clustering algorithms

1 Introduction

Community detection aims to extract set of clusters in large graphs. Each cluster contains a group of nodes which have a high relationship between them. According to the previous studies of Hotho et al. [1], a social network can be represented by undirected hypergraph, called: Folksonomy. In this hypergraph, the nodes represent the set of users, tags and documents. The edges represent the relations that connect these nodes. Several approaches have been developed for the community detection in social networks. Some of these approaches divide the entire graph to many subgraphs, where each subgraph represents a community as in the Spectral algorithm [2] and the division algorithms Radicchi [3]. Other approaches are based on subgroup discovery [4], genetic algorithm [5, 6], random walks on graphs [7–9], Non-negative matrix factorization (NMF) [10], or on the nature of overlapping communities in the real world scenario [11].

Clustering is one of the most used techniques in data analysis process. It aims to regroup the objects into groups, where the similarity between the objects of a same group is higher than the similarity between the objects from different groups. Many studies have exploited the clustering algorithms to identify the contained communities in the network [12–15]. The k-means, the k-medoids and the agglomerative clustering algorithms are most used in this area. In this paper, we aim to evaluate the impact of these three clustering algorithms on the community detection in the academic social bookmarking system: Bibsonomy [16]. To realize that, we have used a sample of Bibsonomy users to construct a distance matrix. Then, we have applied the clustering algorithms on the obtained matrix. To determine the number of clusters, we have used the clustering validation measure: SSE(Sum of squared errors). This measure allows us to determine the ideal number of clusters in the k-means and the k-medoids algorithms. To evaluate the performance of each clustering algorithm, we have used the SSE measure and a human evaluation, which aims to determine the number of similar users in each obtained cluster. The rest of the paper is organized as follows: Sect. 2 provides an overview of the related work in the area of similarity calculation between items. Section 3 presents our method for measuring the distances between users. Section 4 describes the experimental and the computational details. Section 5 describes the experimental results. Finally, we conclude the paper in Sect. 6.

2 Related Work

In social networks, finding the similar items, tweets or users, represents an important step in recommendation systems [19–23], topic detection [17] and community detection [18]. Most of previous recommendation approaches calculate a similarity between items and target users, a similarity between items, or a similarity between target users, to select and recommend the most suitable items to user interests. Kumar and Talebi [19] proposed a new movies recommendation approach, which calculate a similarly between user's tweets and scenarios of movies. They used a variant of k-NN algorithm to predict movie tweets popularity. Zangeler et al. [20] proposed new approach which recommends the appropriate tags to the user during the creation process of the new tweet. Lu et al. [21] re-rank the tweets in the user timeline, by calculating a similarity between the tweets and the user profile. The user's previous tweets are used to construct the user profile. Ricci et al. [22] proposed a new recommendation approach based on the twofold similarity. This approach combines in a novel way content and collaborative-based filtering techniques. Armentano et al. [23] developed a new recommendation algorithm which aims to recommend information sources to information seekers in Twitter. This algorithm selects a set of candidate users and ranks them according to the similarity between the content of published tweets by the candidate users and the target user interests.

In the field of community detection in social network, many studies have exploited the weights of social network edges which relate the social network nodes, to detect the similar nodes. Steinhaeuser and Nitesh [28] proposed a new community detection approach for social network based on edge weighting methods. They calculated the edge weight between nodes using the clustering coefficient similarity (CCS), the Common Neighbor Similarity (CNS) and the Node Attribute Similarity (NAS). Liu et al. [29] proposed a new algorithm based on multiobjective evolutionary algorithm, called MEAsSN. This algorithm can detect both the separated and overlapping communities from signed social networks. Pizzuti [30] proposed a genetic based algorithm to discover communities in social networks. This algorithm introduces the concept of community score and searches for an optimal partitioning of the network by maximizing the community score. Du et al. [11] proposed a new algorithm for community detection in large-scale social networks, called ComTector (Community DeTector). This algorithm is based on the nature of overlapping communities in the real world and does not require any priori knowledge about the number or the original division of the communities. Qi et al. [31] proposed an algorithm for community detection with edge content. This approach exploits the structural and content aspects of the network with the use of a matrix factorization approach.

Other approaches calculate a similarity between users to achieve different goals. Lee et al. [24] proposed a new method for calculating the similarity between users, using the user's location information. They used the semantics of the location rather than the physical location, in order to capture the user's intention and interest. ElSherief et al. [25] proposed a novel temporal similarity metric, based on matrix vectorization, to determine the similar mobile users. Lv et al. [26] proposed a new approach to measure user similarity for location-based social networks (LBSNs). This approach is based on GPS trajectory mining, to extract the routine activities of the users. McKenzie et al. [27] proposed a new approach which exploits sparse and unstructured data provided by other users, to calculate user similarity. In our work, the user similarity is based on the number of common tags with other users. The similarity between two users is equal to the number of common tags they have used to annotate the bookmarks in Bibsonomy. The next section represents the details of our method of calculating the similarities between users.

3 User Similarity Calculation

In our work, the similarity between two users U_1 and U_2 is calculated according to the number of common tags between them. This similarity is calculated as follows:

$$\mathrm{Sim}(U_1, U_2) = \left|\mathrm{tags}_{U_1} \cap \mathrm{tags}_{U_2}\right| / \left|\mathrm{tags}_{U_1} \cup \mathrm{tags}_{U_2}\right| \tag{1}$$

Where:

tags_{U_1}: is the set of used tags by the user U_1 to annotate his bookmarks.
tags_{U_2}: is the set of used tags by the user U_2 to annotate his bookmarks.

This similarity is used to construct the distance matrix M. Each element of the matrix M, represents the distance between two users. This distance is calculated using the following formula:

$$M[i,j] = \begin{cases} 1/\mathrm{Sim}\left(U_i, U_j\right) & \text{if } \mathrm{Sim}(U_i, U_j) > 0 \\ 100000 & \text{if } \mathrm{Sim}(U_i, U_j) = 0 \end{cases} \tag{2}$$

Where:
U_i: is the user i in the distance matrix *M*.
U_j: is the user j in the distance matrix *M*.

Thereafter, the clustering algorithms will be conducted on this distance matrix. These algorithms will allow us to extract the sets of similar users from the Bibsonomy dataset, in order to find the similar communities in this social network.

4 Experimental and Computational Details

4.1 Data Description

In this paper, we will use the 2016-01-01 version of the BibSonomy dataset[1]. This dataset contains three files:

1- The « Tas » file: contains the set of tag assignments. It contains a table that indicates for each user the set of his used tags and bookmarks.
2- The « Bookmark » file: contains information about the bookmarks (The bookmark URL, the uploading date of the bookmark and its description)
3- The « Bibtex » file: contains information about BibTeX data (scientific publications). We have chosen this version of dataset because it contains more information than the old versions. The 2016-01-01 version contains 12296 users and 312584 tags. This large number of users and tags will increase our chance to find various clusters with each clustering algorithm. In our study, we are interested in the « Tas » file, because it contains the set of users and their used tags. This information allows us to calculate the similarities between users and construct the distance matrix. To provide useful data for our evaluation methods, the distance matrix will be constructed using the most popular tags and the most popular users.

4.2 Descriptive Statistics of the Used Dataset

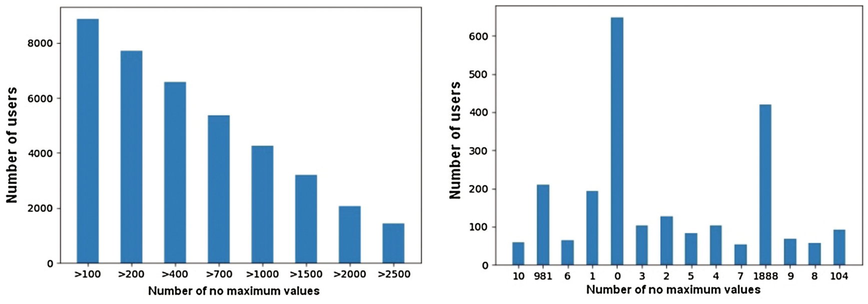
In this section we give a descriptive statistics of the « Tas » file. These statistics are obtained using some techniques of data visualization, such as: the self-organizing map (SOM) and the bar graph. These techniques will be applied on the distance matrix and the inverted file, where its rows represent the set of users and its columns represent the set of tags used by these users.

4.2.1 Bar Graphs of no Maximum Values

To predict the number of similar communities in Bibsonomy dataset, we have computed the number of no maximum values in the distance matrix. Then, we have created the bar graphs from this matrix. These bar graphs will show us the distribution of users according to their number of relations with others users. Figure 1 represents the

[1] Knowledge and Data Engineering Group, University of Kassel: Benchmark Folksonomy Data from BibSonomy, version of January 01st, 2016. http://bibsonomy.org/.

obtained bar graphs. From Fig. 1(b) we can see that more than 600 users don't have any relations (The 0 value). We can see also, that more than 400 users have only 1889 relations in total and less than 100 users have only 10 relations. From Fig. 1(a) we can see that more than 8000 users have relations with more than 100 users, and less than 2000 users have relations with more than 2500 users. The obtained information from these bar graphs indicates that there is a high probability to find similar users among the set of users which have more relations with other users. Hence, we have decided to use only the most popular users to construct the distance matrix in the clustering step.

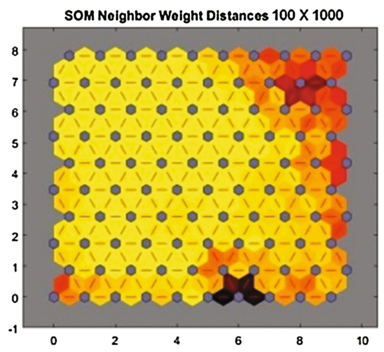


(a) The ranges of values (b) The exact values

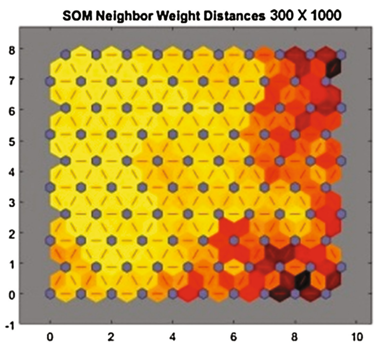
Fig. 1. The bar graphs of no maximum values. It represents a sample of no maximum values. The x-axis in (b) represent the exact values, where in (a) represent the ranges of values.

4.2.2 Self-Organizing Map (SOM)

The self-organizing map is generally used to map high-dimensional data into a two-dimensional representation. SOM is a data visualization technique which reduces the dimensions of data through the use of self-organizing neural networks. In our case, we have constructed two inverted files from a sample of 100 and 300 most popular users. To decrease the number of zero values in the inverted files, we have used the set of the 1000 most used tags by these users. Then, we have applied the SOM on these inverted files. Figure 2 show the unified distance matrix (U-Matrix) in each inverted file. We believe that we can detect diverse communities when we observe diverse colors, like in Fig. 2 (b). This is due to the disparity of users' relations number in each inverted file. This number is high when users use the same tags to annotate the documents. Figure 2 can't give us an estimation of similar users groups, but it can help us to predict the number of these groups.



(a) 100 popular users



(b) 300 popular users

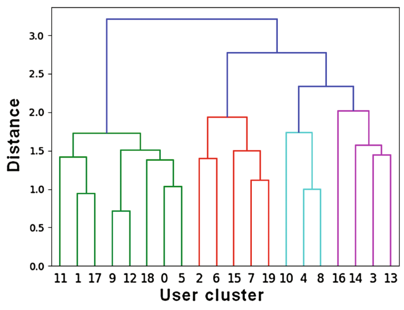
Fig. 2. The U-Matrix of inverted files. The U-Matrix (a) and (b) are obtained respectively from a sample of 100 and 300 users.

4.3 Methodology Description

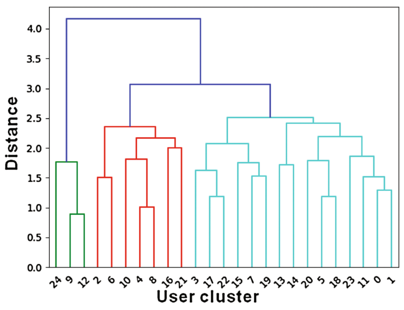
In this section, we describe our methodology of applying the clustering algorithms on Bibsonomy dataset in order to extract the set of similar users.

4.3.1 Detection of the Similar Users Using the Agglomerative Clustering

Agglomerative clustering algorithm consists of building a hierarchy of clusters, by merging the pairs of clusters until all clusters have been merged into one single cluster that contains all documents. In our case, these clusters should contain a set of Bibsonomy users. To realize that, we have constructed the distance matrix that measures the distances between users. According to theses distances, the users will be separated into different groups. The distance between two users is small when they have similar centers of interests and large when they haven't any similar centers of interests. This distance is calculated using the formula (2). Two users are similar, when they have tags in common. Figure 3 shows the obtained dendrograms using the agglomerative clustering algorithm on a set of 20 and 25 users from the entire set of users.



(a) 20 users

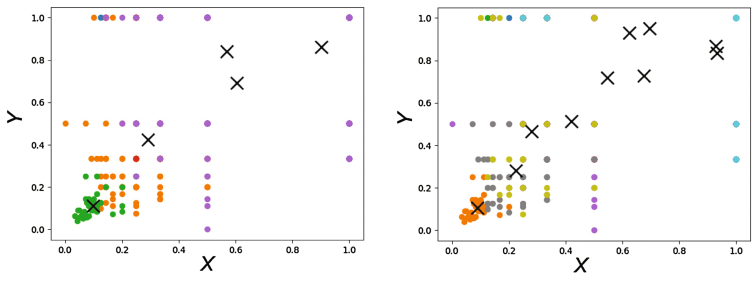


(b) 25 user

Fig. 3. Obtained dendrograms using agglomerative clustering.

4.3.2 Detection of the Similar Users Using the k-Means and the k-Medoids

The k-means algorithm consists of grouping data into k clusters. The center of each cluster represents the mean point of all data points which belong to this cluster. Each data point in our case represents a Bibsonomy user. The k-medoids is similar to the k-mean algorithm. However the centers of clusters in the k-medoids algorithm are assumed as points from the set of the data points. The k-medoids attempts to minimize the sum of dissimilarities between the data points of a cluster and the point designated as the center of that cluster. The k-means and the k-medoids algorithms will be conducted on users' distance matrix. Figure 4 shows the k-means results with 5 and 10 clusters using a set of 100 users.



(a) 5 clusters (b) 10 clusters

Fig. 4. K-means clustering results.

4.4 Determination of the Best Number of Clusters

The ideal number of clusters remains an important information to make this comparison. To realize that, we have used the sum of squared errors (SSE) to detect the ideal number of clusters in k-means algorithm. This number will be used also in k-medoids and agglomerative clustering (to determine the cut-off point of the dengoram). Figure 5 shows the SSE of the k-means algorithm using a set of 100 users.

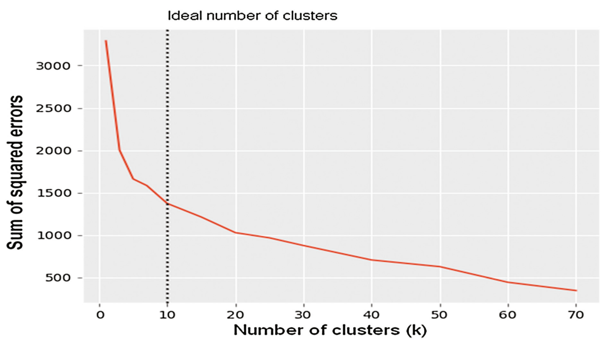


Fig. 5. Line graph of the (SSE) versus number of clusters using k-means. The ideal number of clusters for this dataset is 10.

5 Experimental Results

In this section, we will compare three clustering algorithms: the agglomerative clustering, the k-means and the k-medoids for similar users' detection in Bibsonomy. We will apply these algorithms on the distance matrix, which is constructed from a set of 100 users. To evaluate each approach, we will use (1) the SSE measure and (2) a human evaluation. The human evaluation allows us to understand the meaning of the obtained clusters (i.e. if the clusters are well made or not). We suppose that the best clustering algorithm should determine various clusters. These clusters should have a low SSE value and contain a high number of similar users. According to these criteria, the clustering algorithms will be compared. Two users are considered similar if they have similar topics.

5.1 Obtained Results

In our first comparison, we calculate for each obtained cluster, the number of similar users. The clustering algorithms will be compared according to the average number of similar users in the whole obtained clusters. Table 1 shows the obtained results with each clustering algorithm using a set of 100 users. In Table 1 we can see that the average number of similar users when using the k-means algorithm is higher than the one obtained with the other algorithms. Our second comparison is based on the SSE measure. Table 2 shows the SSE values with each clustering algorithm. From Table 2, we can see that the k-means have the lowest SSE values in most cases. These results indicate that the quality of the obtained clusters with k-means is better than the ones obtained with the other algorithms. We conclude that the k-means algorithm is the most suitable to detect the similar communities in Bibsonomy social network.

Table 1. Number of similar users with each clustering algorithm.

Algorithm	cluster 1	cluster 2	cluster 3	cluster 4	cluster 5	cluster 6	cluster 7	cluster 8	cluster 9	cluster 10	Average
k-means	3	8	2	4	19	21	0	0	2	4	6
k-medoids	2	5	7	7	6	2	9	3	5	2	5
Agglomerative	4	0	0	3	0	14	16	0	0	7	4

Table 2. SSE values in each clustering algorithm.

Algorithm	k = 3	k = 5	k = 7	k = 10	k = 15	k = 20	k = 25	k = 30	k = 40	k = 50	k = 60	Average
k-means	2002	1642	1600	1412	1240	1093	992	905	750	596	480	1155
k-medoids	2002	1688	1629	1541	1358	1094	938	917	724	624	490	1182
Agglomerative	2002	1715	1527	1539	1321	1200	1070	911	740	573	457	1186

6 Conclusion

In this paper we have studied the impact of the clustering algorithms for the community detection in Bibsonomy. The obtained results show that the k-means algorithm performs better than the k-medoids and the agglomerative clustering algorithms for community detection in Bibsonomy. In our future work, we aim to propose a new recommendation approach which can help the Bibsonomy users to find relevant bookmarks to their centers of interests. This recommendation approach will be based on the k-means algorithm.

References

1. Hotho, A., Jäschke, R., Schmitz, C., Stumme, G.: Information retrieval in folksonomies: search and ranking. In: European Semantic Web Conference, pp. 411–426. Springer, Heidelberg, June 2006
2. Newman, M.E.: Modularity and community structure in networks. Proc. Natl. Acad. Sci. **103** (23), 8577–8582 (2006)
3. Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., Parisi, D.: Defining and identifying communities in networks. Proc. Natl. Acad. Sci. U.S.A. **101**(9), 2658–2663 (2004)
4. Atzmueller, M., Doerfel, S., Mitzlaff, F.: Description-oriented community detection using exhaustive subgroup discovery. Inf. Sci. **329**, 965–984 (2016)
5. Pizzuti, C.: Ga-net: a genetic algorithm for community detection in social networks. In: International Conference on Parallel Problem Solving from Nature, pp. 1081–1090. Springer, Heidelberg, September 2008
6. Tasgin, M., Herdagdelen, A., Bingol, H.: Community detection in complex networks using genetic algorithms. arXiv preprint arXiv:0711.0491 (2007)
7. Pons, P., Latapy, M.: Computing communities in large networks using random walks. In: International Symposium on Computer and Information Sciences, pp. 284–293. Springer, Heidelberg, October 2005
8. Gaume, B.: Balades aléatoires dans les petits mondes lexicaux. I3 Inf. Interact. Intell. **4**(2), 39–96 (2004)
9. Zhou, H., Lipowsky, R.: Network brownian motion: a new method to measure vertex-vertex proximity and to identify communities and subcommunities. In: International Conference on Computational Science, pp. 1062–1069. Springer, Heidelberg, June 2004
10. Wang, F., Li, T., Wang, X., Zhu, S., Ding, C.: Community discovery using nonnegative matrix factorization. Data Min. Knowl. Disc. **22**(3), 493–521 (2011)
11. Du, N., Wu, B., Pei, X., Wang, B., Xu, L.: Community detection in large-scale social networks. In: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis, pp. 16–25. ACM, August 2007
12. Zhou, K., Martin, A., Pan, Q., Liu, Z.G.: Median evidential c-means algorithm and its application to community detection. Knowl.-Based Syst. **74**, 69–88 (2015)
13. Zhang, S., Wang, R.S., Zhang, X.S.: Identification of overlapping community structure in complex networks using fuzzy c-means clustering. Physica A **374**(1), 483–490 (2007)
14. Jiang, Y., Jia, C., Yu, J.: An efficient community detection method based on rank centrality. Physica A **392**(9), 2182–2194 (2013)
15. Estrada, E.: Community detection based on network communicability. Chaos: An Interdisciplinary. J. Nonlinear Sci. **21**(1), 016103 (2011)

16. Benz, D., Hotho, A., Jäschke, R., Krause, B., Mitzlaff, F., Schmitz, C., Stumme, G.: The social bookmark and publication management system bibsonomy. VLDB J.—Int. J. Very Large Data Bases [1] **19**(6), 849–875 (2010)
17. Elbagoury, A., Ibrahim, R., Farahat, A.K., Kamel, M.S., Karray, F.: Exemplar-based topic detection in twitter streams. In: ICWSM, pp. 610–613, April 2015
18. Zhou, K., Martin, A., Pan, Q.: A similarity-based community detection method with multiple prototype representation. Physica A **438**, 519–531 (2015)
19. Peska, L., Vojtas, P.: Hybrid Biased k-NN to Predict Movie Tweets Popularity (2015)
20. Zangerle, E., Gassler, W., Specht, G.: Recommending#-tags in twitter. In: Proceedings of the Workshop on Semantic Adaptive Social Web (SASWeb 2011). CEUR Workshop Proceedings, vol. 730, pp. 67–78 (2011)
21. Lu, C., Lam, W., Zhang, Y.: Twitter user modeling and tweets recommendation based on wikipedia concept graph. In: Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 2012
22. Ricci, F., Venturini, A., Cavada, D., Mirzadeh, N., Blaas, D., Nones, M.: Product recommendation with interactive query management and twofold similarity. In: International Conference on Case-Based Reasoning, pp. 479–493. Springer, Heidelberg, June 2003
23. Armentano, M.G., Godoy, D., Amandi, A.A.: Recommending information sources to information seekers in twitter. In: International Workshop on Social Web Mining, July 2011
24. Lee, M.J., Chung, C.W.: A user similarity calculation based on the location for social network services. In: International Conference on Database Systems for Advanced Applications, pp. 38–52. Springer, Heidelberg, April 2011
25. ElSherief, M., ElBatt, T., Zahran, A., Helmy, A.: The quest for user similarity in mobile societies. In: 2014 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), pp. 569–574. IEEE, March 2014
26. Lv, M., Chen, L., Chen, G.: Mining user similarity based on routine activities. Inf. Sci. **236**, 17–32 (2013)
27. McKenzie, G., Adams, B., Janowicz, K.: A thematic approach to user similarity built on geosocial check-ins. In: Geographic information science at the heart of Europe, pp. 39–53. Springer (2013)
28. Steinhaeuser, K., Chawla, N.V.: Community detection in a large real-world social network. Social Computing, Behavioral Modeling, and Prediction, pp. 168–175 (2008)
29. Liu, C., Liu, J., Jiang, Z.: A multiobjective evolutionary algorithm based on similarity for community detection from signed social networks. IEEE Trans. Cybern. **44**(12), 2274–2287 (2014)
30. Pizzuti, C.: GA-Net: a genetic algorithm for community detection in social networks. In: PPSN, pp. 1081–1090, September 2008
31. Qi, G.J., Aggarwal, C.C., Huang, T.: Community detection with edge content in social media networks. In: 2012 IEEE 28th International Conference on Data Engineering (ICDE), pp. 534–545. IEEE, April 2012

Big Data, Knowledge Discovery and Data Mining, Knowledge Based Management

Using Predictive Data Mining Models for Data Analysis in a Logistics Company

Miroslava Muchová(✉), Ján Paralič, and Michael Nemčík

Department of Cybernetics and Artificial Intelligence,
Faculty of Electrical Engineering and Informatics,
Technical University of Košice, Košice, Slovak Republic
{miroslava.muchova, jan.paralic}@tuke.sk,
michael.nemcik@student.tuke.sk

Abstract. The aim of this paper is to apply predictive data mining (DM) techniques in order to predict the average fuel consumption for trucks and drivers resp., to identify the key factors that affect fuel consumption of vehicles and also to identify best practices and driving styles of drivers. For this purpose different models have been proposed to provide an overview of the key factors affecting fuel consumption for individual vehicles and their drivers. Predictive models enabled us to identify main influencing factors and provide recommendations for a logistics company to reduce the fuel consumption. Data were collected from Dynafleet information system of a small transport company. The company is dealing with freight traffic, particularly trucks. We first describe selected projects dealing with similar tasks in this area. Next, we explore and analyze data using CRISP-DM methodology by appropriate methods designed for data mining and then evaluate the results of the experiments.

Keywords: Data mining · CRISP-DM · Predictive data mining · Fuel consumption · Dynafleet · Naive bayes · Neural network

1 Introduction

Countless data is collected in an important sector of the economy, the logistics. It is not just about customers and their supermarket shopping. In order for the products to pass through the cash register, it is necessary to perform the next steps related thereto. From shipping planning and tracking, delivery and stock management to replenishment of shelves in stores [1]. For the transport of such shipments, trucks are used most often and provide an unmatched quantum of data through various devices, which can then be explored by data mining [2].

Predictive data mining methods are aimed at finding and describing previously unknown and non-trivial relationships and data contexts [3]. The data that were used in this article comes from a logistics company. The company deals with freight transport, namely trucks. Truck, semi-trailers and heavy-duty trailers are now an indispensable part of the freight. Every truck operation has as it is a primary goal to save money or, reduce company costs [4, 5]. Drivers' driving skills and road conditions have a significant impact on fuel consumption and vehicle wear. There are activities that apply properly to drivers, reduce fuel consumption, reduce the load on some parts, and reduce emissions.

2 Related Work

One of the goals described in the article [6] was predicting the average fuel consumption for a specific vehicle and route. Vehicle fuel consumption models were created based on different variables, for example, driving behavior, technical characteristics of the vehicle, road characteristics and weather data.

In order to create meaningful models and evaluate fuel consumption, it was necessary to combine data from different sources. First, the fuel-efficient vehicle data was paired for every 60 s and then every 10 min. This resulted in a 10% dilution of the data from the original dataset. These alternative data sets were used to train models to assess whether a 10-minute frequency is sufficient to build a usable fuel consumption prediction.

Data from other sources was also paired. After removing the missing values, from the original 5 million only 2.7 million entries remained.

Prior to model training, the data was divided into a training, validation and test set. The training set was used to train regression models. The validation set was used to avoid classifier overfitting. Files were split by date. According to this distribution, approximately 67% of the data was included in the training, 11% in the validation and 22% in the test set.

The results demonstrate that 10-minute datasets show a lower error rate than one minute. For this reason, it would be more interesting to look at predictive models with fewer samples to reduce the error rate. The results also showed that vehicle weight, vehicle speed, slope, direction and wind speed are some of the most important variables for all models.

Another case study [7] dealt with the impact of the driving style of individual drivers on fuel consumption. The authors in this study used data from public transport in Lisbon. The data was automatically retrieved from the CAN bus for three years. The aim was to apply knowledge discovery techniques and identify the main factors that affect the average fuel consumption, classify drivers by performance and identify the most appropriate driving techniques and styles.

Subsequently, the data was divided into five classes with approximately the same number of events. The result of this process was the distribution of fuel consumption into five subclasses from 20 to 110 L per 100 km.

After the discretization process, the investigators used the Naive Bayes 14technique to determine the main factors that could have a major impact on fuel consumption. The main factors most affecting the reduction in fuel consumption were: reduce clutch usage, maximize time spent in inertia, minimize excessive engine speeds, avoid traffic jams.

70% of the data was used for training and 30% of the data for testing. The accuracy of the resulted model was lower in higher fuel consumption classes, mainly because 92.3% of the drivers were in the first three classes. The class with the lowest fuel consumption had the highest (90%) success rate.

The results show that fuel consumption can be reduced by 3 to 5 L per 100 km in a suitable style, which is between € 20 and € 40 per day and € 1.5 million a year.

3 Methodology

We used the CRISP-DM methodology [8]. CRISP-DM is a data mining process model [12] that describes commonly used approaches for data mining projects independent from industry sector and technology used. CRISP-DM consists of six phases: Business understanding, Data understanding, Data preparation, Modeling, Evaluation, and Deployment.

Business understanding: Our aim was to process the acquired data from given logistics company dealing with freight transport and analyze it by means of DM for better freight management.

Data understanding: The data which we used comes from the Dynafleet Online system. This is a web application that allows users to track their vehicles. This system allows to collect data from every truck, which travels all around the Europe [9]. The data used to create predictive models was collected between 2013 and 2016 (from June 21, 2013 to September 30, 2016). The company operates six Volvo trucks and employs 18 drivers (only men). The data which we analyzed concerns 14 drivers driving in 19 European countries [10].

Vehicle data come from two different types of data files. The first file type contains evaluated fuel consumption data, and the second type of file contains specific vehicle data for the selected time interval.

Data preparation: Firstly, we created a database and filled it with the data. We worked with database platform SQL Server 2014 from Microsoft. SQL Server 2014 is a complex and sophisticated system with a simple user interface that enables managers to benefit from the concept of "self-service DM."

All attributes (Coasting time, Brake/stop relation, Above-economy time, Overspeed, Time at idle, etc.) are numeric except the Driver name attribute, which is symbolic. Attributes representing performance are expressed in the system as percentage values:

- 80–100 = Good
- 60–79 = Average
- 0–59 = Improvable

The total score includes prediction, standstill, speed adaptation, engine and gear utilization. The total score is a way of assessing the driver's driving style over a given period of time. Most of the data we had available was first discretized (for details see next section).

4 Experiments and Their Evaluation

Modeling: Using selected DM techniques, we analyzed the fuel consumption. In the first three experiments, we decided to use the Naive Bayes technique because it was very effective in one of the analyzed studies described above [7], and that was the main reason for choosing this technique. In the last experiment, we used the Neural Network technique, which can also be seen as a generalization of Bayesian approach to classification [13]. The Microsoft Neural Network implementation in MS Visual Studio 2013 uses a Multilayer Perceptron network, which consists of three layers. This is an input layer, a hidden layer, and an output layer. This type of neural network works with continuous and discrete values. It is also able to solve non-linear problems. We used the default parameter settings of this module.

4.1 Experiment 1

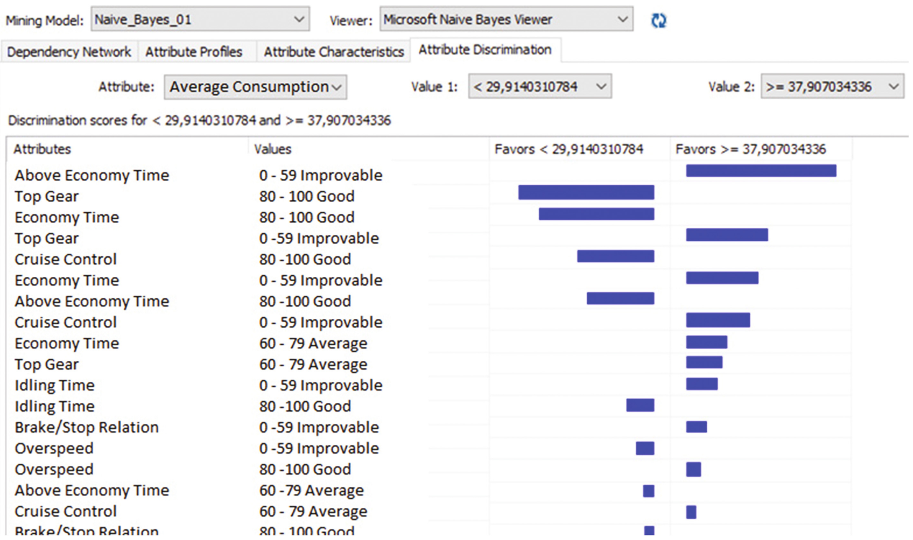
The aim of this experiment was to identify the key factors that have the greatest impact on fuel consumption.

We used the automatic discretization functionality and three fuel consumption classes were created:

- bellow 29.91 l
- from 29.91 l to 37.90 l
- above 37.90 l

We also used automatic discretization functionality for vehicle weight:

- bellow 21 556 kg
- from 21 556 kg to 33 851 kg


Mining Model: Naive_Bayes_01 Viewer: Microsoft Naive Bayes Viewer

Dependency Network | Attribute Profiles | Attribute Characteristics | Attribute Discrimination

Attribute: Average Consumption Value 1: < 29,9140310784 Value 2: >= 37,907034336

Discrimination scores for < 29,9140310784 and >= 37,907034336

Attributes	Values	Favors < 29,9140310784	Favors >= 37,907034336
Above Economy Time	0 - 59 Improvable		■
Top Gear	80 - 100 Good	■	
Economy Time	80 - 100 Good	■	
Top Gear	0 -59 Improvable		■
Cruise Control	80 -100 Good	■	
Economy Time	0 - 59 Improvable		■
Above Economy Time	80 -100 Good	■	
Cruise Control	0 - 59 Improvable		■
Economy Time	60 - 79 Average		■
Top Gear	60 - 79 Average		■
Idling Time	0 - 59 Improvable		■
Idling Time	80 -100 Good	■	
Brake/Stop Relation	0 -59 Improvable		■
Overspeed	0 -59 Improvable	■	
Overspeed	80 -100 Good		■
Above Economy Time	60 -79 Average	■	
Cruise Control	60 - 79 Average		■
Brake/Stop Relation	80 - 100 Good	■	

Fig. 1. Key factors to improve fuel efficiency for two classes of fuel consumption

- above 33 851 kg

In Fig. 1, we can see calculated discrimination scores between two of the classes of fuel consumption. If a blue bar appears on the right or left hand side, it means that the attribute value is dependent on that class. Using this analysis, we can determine which attributes the drivers should “improve” in order to achieve lower fuel consumption.

If we focus on the highest fuel consumption class, we can see that the highest probabilities have attributes that have been rated from 0 to 59. The highest fuel consumption class is most influenced by the following four attributes:

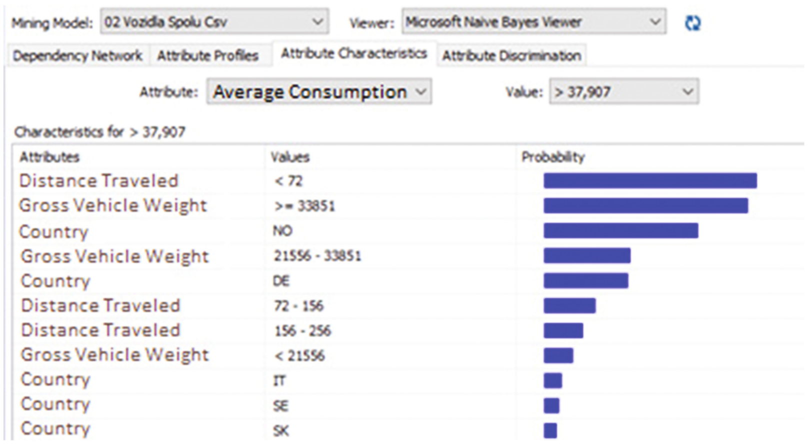
- Above economy time
- Top gear
- Economy time
- Cruise control

Table 1 shows the confusion matrix of the results, with the predicted class in rows and actual class in columns. The overall accuracy of the model is 74.74%. The highest accuracy is achieved for the last class (79%) followed by the first class (69%).

Table 1. Confusion matrix (values are given in percentage)

Predicted class	Actual class		
	<29.91	29.91–37.90	>37.90
<29.914	69%	28%	3%
29.91–37.90	44%	50%	6%
>=37.90	0%	21%	79%

In Fig. 2 we can see that drivers with high fuel consumption (above 37.901) in most cases traveled less than 72 km, the weight of their vehicle was over 33.851 tons and drivers ride mostly in Norway.


Mining Model: 02 Vozidla Spolu Csv
Viewer: Microsoft Naive Bayes Viewer
Dependency Network
Attribute Profiles
Attribute Characteristics
Attribute Discrimination

Attribute: Average Consumption
Value: > 37,907
Characteristics for > 37,907

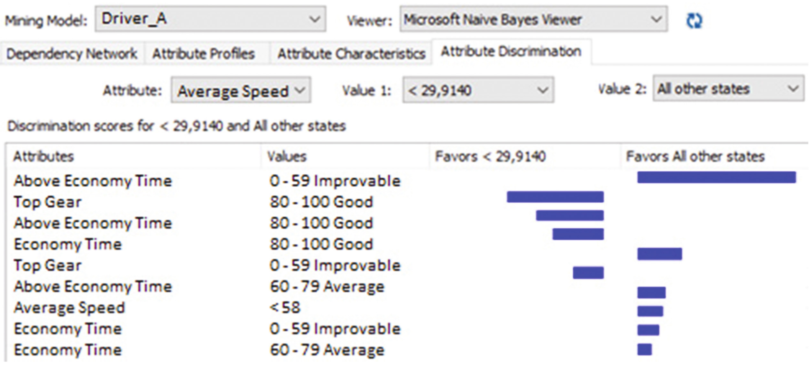
Attributes	Values	Probability
Distance Traveled	< 72	
Gross Vehicle Weight	>= 33851	
Country	NO	
Gross Vehicle Weight	21556 - 33851	
Country	DE	
Distance Traveled	72 - 156	
Distance Traveled	156 - 256	
Gross Vehicle Weight	< 21556	
Country	IT	
Country	SE	
Country	SK	

Fig. 2. Key factors to improve fuel efficiency for the high fuel consumption class

4.2 Experiment 2

In this experiment we have created models that provide an overview of key factors affecting fuel consumption for individual drivers. The main aim of this experiment was to identify these factors and provide drivers recommendations that would lead to reduction in their fuel consumption. The analyses were conducted for drivers whose total mileage was over than 60,000 km.

Figures 3 and 4 show results of the analysis of drivers A and G, respectively. We can see the main factors that mostly affect the fuel consumption for these drivers.



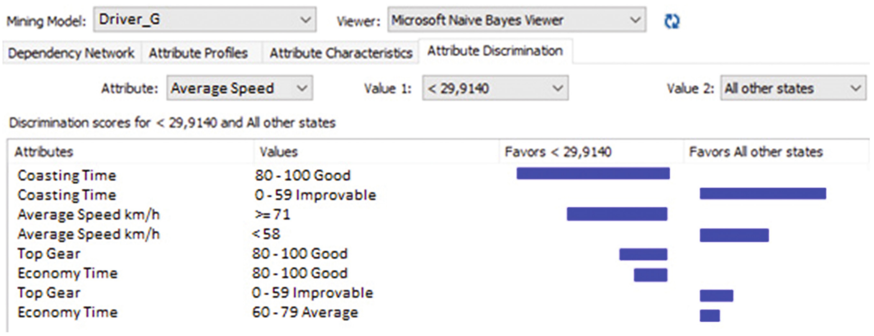
Attribute: Average Speed Value 1: < 29,9140 Value 2: All other states

Discrimination scores for < 29,9140 and All other states

Attributes	Values	Favors < 29,9140	Favors All other states
Above Economy Time	0 - 59 Improvable		■
Top Gear	80 - 100 Good	■	
Above Economy Time	80 - 100 Good	■	
Economy Time	80 - 100 Good	■	
Top Gear	0 - 59 Improvable		■
Above Economy Time	60 - 79 Average	■	
Average Speed	< 58		■
Economy Time	0 - 59 Improvable		■
Economy Time	60 - 79 Average		■

Fig. 3. Analysis of Driver_A

The analysis for Driver_A shows a high probability of above economy time. This means that if the driver reaches a rating from 0 to 59 for this attribute, there is a high probability that the driver will achieve high fuel consumption.



Attribute: Average Speed Value 1: < 29,9140 Value 2: All other states

Discrimination scores for < 29,9140 and All other states

Attributes	Values	Favors < 29,9140	Favors All other states
Coasting Time	80 - 100 Good	■	
Coasting Time	0 - 59 Improvable		■
Average Speed km/h	>= 71	■	
Average Speed km/h	< 58		■
Top Gear	80 - 100 Good	■	
Economy Time	80 - 100 Good	■	
Top Gear	0 - 59 Improvable		■
Economy Time	60 - 79 Average		■

Fig. 4. Analysis of Driver_G

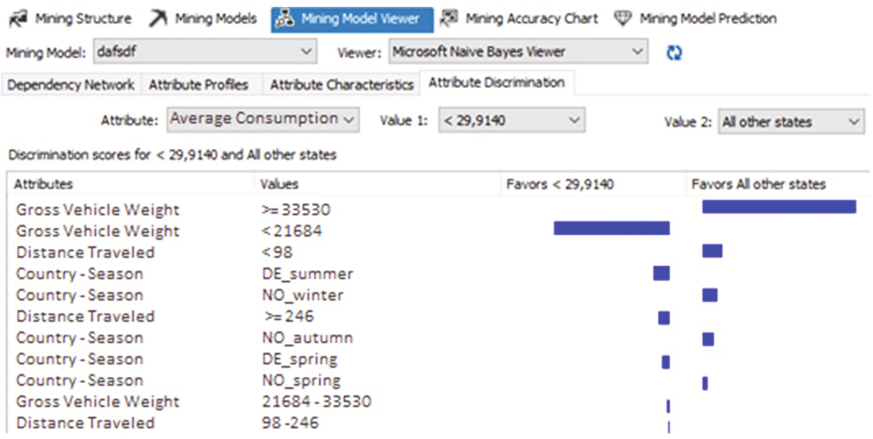
Driver_G has the highest probability of coasting time and a lower probability of attributes as average speed, top gear and economy time.

4.3 Experiment 3

The purpose of this experiment was to determine if seasons and traffic peaks have some influence on fuel consumption. The seasons were divided into spring, summer, autumn, winter, and daytime periods of high traffic were defined from 6:00 to 9:00 and from 15:00 to 18:00.

The dependence between traffic peak and fuel consumption has not been demonstrated, so this attribute has no significant effect on fuel consumption.

Figure 5 shows that winter in Norway has the highest dependency among the season attributes. Somewhat less dependence have autumn and spring, also in Norway. Norway is a country where snow often falls in winter and in autumn it often rains, so it is assumed that due to bad weather, drivers could have higher fuel consumption in these seasons. Low fuel consumption should be achieved by drivers in the summer and spring in Germany.



Mining Model: dafsdf
Viewer: Microsoft Naive Bayes Viewer

Attribute: Average Consumption
Value 1: < 29,9140
Value 2: All other states

Discrimination scores for < 29,9140 and All other states

Attributes	Values	Favors < 29,9140	Favors All other states
Gross Vehicle Weight	>= 33530		■
Gross Vehicle Weight	< 21684	■	
Distance Traveled	< 98		■
Country - Season	DE_summer	■	
Country - Season	NO_winter		■
Distance Traveled	>= 246	■	
Country - Season	NO_autumn		■
Country - Season	DE_spring	■	
Country - Season	NO_spring		■
Gross Vehicle Weight	21684 - 33530	■	
Distance Traveled	98 - 246	■	

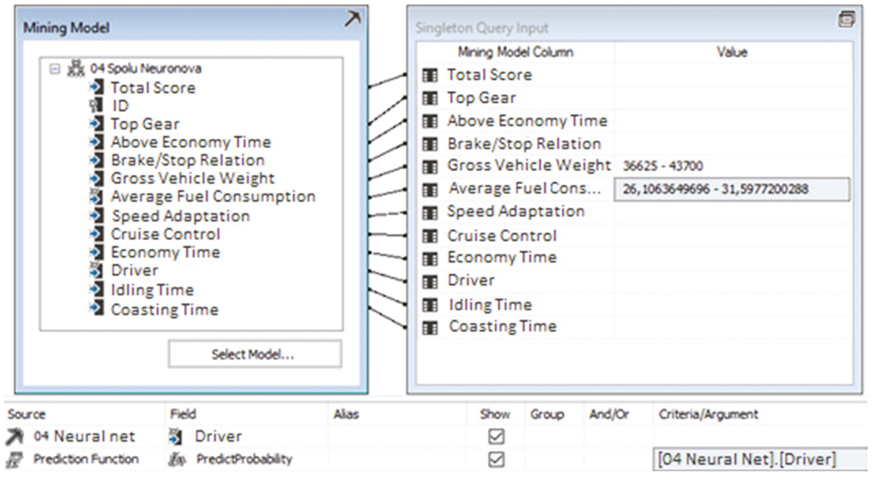
Fig. 5. Comparing the lowest fuel consumption class compared to other classes

4.4 Experiment 4

The aim of the fourth experiment was to develop a model that would recommend the most appropriate driver based on the criteria (route, country, weight of the vehicle…) given and would also predict the average fuel consumption for individual drivers.

After creating the model, we used the Singleton Query option to specify the attributes (average vehicle weight and drivers) [11]. With function Predict we calculated the average fuel consumption that drivers should achieve.

We specified the weight of the vehicle in the range from 36 625 to 43 700 kg and we chose the Driver_J (see the setup presented in Fig. 6).

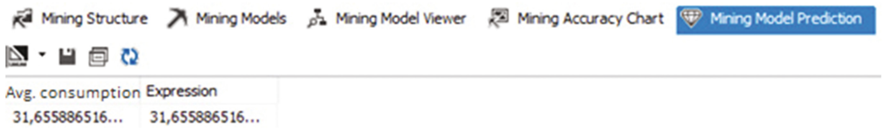


Mining Model Column	Value
Total Score	
Top Gear	
Above Economy Time	
Brake/Stop Relation	
Gross Vehicle Weight	36625 - 43700
Average Fuel Cons...	26,1063649696 - 31,5977200288
Speed Adaptation	
Cruise Control	
Economy Time	
Driver	
Idling Time	
Coasting Time	

Source	Field	Alias	Show	Group	And/Or	Criteria/Argument
04 Neural net	Driver		✓			
Prediction Function	PredictProbability		✓			[04 Neural Net].[Driver]

Fig. 6. Overview of selected specifications

In Fig. 7 we can see that Driver_J should have an average fuel consumption of about 31.66 L per 100 km.



Avg. consumption	Expression
31,655886516...	31,655886516...

Fig. 7. Fuel consumption of driver J

Evaluation: In the same way, we calculated the predicted fuel consumption for all drivers. The worst driver had a fuel consumption of about 36.8 L per 100 km. Average fuel consumption is 33.69 L. If we choose the right driver, the company should save about 2 to 5 L per 100 km. Drivers ride a daily average of almost 500 km, meaning that the company could save about 10 to 25 L a day per vehicle per day.

Deployment: The company currently uses 6 vehicles. After the correct implementation of the project, average fuel consumption per vehicle per day would be reduced by an average of 3.5 L per 100 km per day, i.e. 17 L per day.

The average price of the fuel consumption in 2016 was 1 038€. One vehicle is in operation for around 281 days per year, which means that about 4.777 L can be saved per year on a single vehicle i.e. 4 959€. The company could save about 29 751€ per year.

5 Conclusions

In this paper we presented our analytical work which we performed on real data from a small logistic company in order to improve their transportation processes in terms of reduced costs of their transportation processes. We used CRISP-DM methodology starting from the problem understanding, through data understanding and preparation up to the modeling, evaluation on proposition for deployment of discovered knowledge. In this paper we focused on the results of the modeling phase describing of 4 experiments and their results. Using the Naive Bayesian technique, we determined the factors that have the highest effect on fuel consumption, and by the Neural Network, we determined the most appropriate driver for particular route on the basis of the specified criteria. Finally, we evaluated the experiments also in terms of potential cost savings which could be achieved when discovered knowledge will be deployed in the company's decision processes.

Further research will focus on extending the model to other factors that influence decision making on driver choices. Based on the requirements of the logistics company owner, as well as based on interviews with drivers and knowledge gained from our analyzes, we propose a multi-criteria decision model using proposed driver and delivery route models. The main objective of the project will be to design and create a decision support system for assigning drivers to supply routes. With the consent of the business owner, we will test this model and evaluate to what extent this system is beneficial to the chosen logistics company.

Acknowledgment. The work presented in this paper was partially supported by the Research Grant Agency of the Ministry of Education and Academy of Science of the Slovak Republic under grant no. 1/0493/16 and by the Slovak Cultural and Educational Grant Agency of the Ministry of Education, Science, Research and Sport of the Slovak Republic under the grant No. 025TUKE-4/2015.

References

1. Pour, J., Maryška, M., Novotný, O.: Business Intelligence in business practice (in Czech). PBtisk Přibram (2012)
2. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques, 3rd edn. Morgan Kaufmann, Burlington (2011)
3. Congna, Q., Huifeng, Z.: Study on application of data mining technology to modern logistics management decision. In: International Forum on Information Technology and Applications, pp. 433–436 (2009)
4. Paul A., Saravanan, V.: Data mining analytics to minimize logistics cost. Int. J. Adv. Sci. Technol. 89–107 (2011)
5. Gustin, C.M., Stank, T.P.: Computerization: supporting integration. Int. J. Phys. Distrib. Logist. Manage. **24**(1), 11–16 (2006)
6. Laxhammar, R., Gascón-Vallbona, A.: Vehicle models for fuel consumption. EC FP6 project COMPANION deliverable D4.3 (2015)

7. Ferreira, J., Almeida, J., Silva, A.: The impact of driving styles on fuel consumption: a data warehouse and data mining based discovery process. IEEE Trans. Intell. Transp. Syst. **16**(5), 2653–2662 (2015)
8. Paralič, J.: Knowledge Discovery in Databases. Elfa, Košice (2003). (in Slovak)
9. Muchová, M., Paralič, J.: Analyzing data to improve a logistics company's specific business process. In: WIKT&DaZ 2016: Proceedings from Conference. Bratislava: STU, pp. 299–304 (2016). (in Slovak)
10. Muchová, M: Big data analysis in selected logistics process. In: SCYR 2016: Proceedings from Conference, Košice: TU, pp. 55–57 (2016)
11. Uldrich, M., Jurczyk T.: Neural networks and their use. IT Systems, No. 3 (2014). (in Czech)
12. Wirth, R., Hipp, J.: CRISP-DM: towards a standard process modell for data mining. In: Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining (2000)
13. Ruck, D.W., Rogers, S.K., Kabrisky, M., Oxley, M.E., Suter, B.W.: The multilayer perceptron as an approximation to a Bayes optimal discriminant function. IEEE Trans. Pattern Anal. Mach. Intell. **13**, 163–174 (1991)
14. Rish, I.: An empirical study of the Naive Bayes classifier. In: IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence, vol. **22**(3). IBM (2001)

Twitter Sentiment Analysis Using a Modified Naïve Bayes Algorithm

Manav Masrani and Poornalatha G.(✉)

Department of Information and Communication Technology MIT,
Manipal University, Manipal, India
manav.masrani@gmail.com, poornalatha.g@manipal.edu

Abstract. Microblogging has emerged as a popular platform and a powerful communication tool among people nowadays. A clear majority of people share their opinions about various aspects of their lives online every day. Thus, microblogging websites offer rich sources of data in order to perform sentiment analysis and opinion mining. Because microblogging has emerged relatively recently there are only some research works which are devoted to this field. In this paper, the focus is on performing the task of sentiment analysis using Twitter which is one of the most popular microblogging platforms. Twitter is a very popular microblogging site where its users write status messages called tweets to express themselves. These status updates mostly express their opinions about various topics. The objective of this paper is to build a system that can classify these Twitter status updates as positive, negative, or neutral with respect to any query term thereby giving an idea about the overall sentiment of the people towards that topic. This type of sentiment analysis is useful for advertisers, consumers researching a service or product, companies, governments, marketers, or any organization who are researching public opinion.

Keywords: Twitter · Data mining · Sentiment analysis

1 Introduction

Social networking sites have received more attention than ever nowadays. A clear majority of people use the platform of social media to express and spread their opinions continually about a variety of topics. With the rise in use of micro-blogging sites like Twitter, people can express and share their opinions with each other on a common platform. Microblogging sites such as Twitter provide rich sources of information about people, products, personalities, and trends etc.

Twitter is one example of the impact that social media has on today's society. With continuously increasing popularity, Twitter has become a frontrunner in social networking, second only to Facebook. The primary advantage of Twitter over other social media websites is its 140-character limit on Tweets. Thus, the status updates or tweets posted by Twitter users are precise and to the point as compared to other social media sites.

People continuously use Twitter to express themselves or to share advice, news, facts, concerns, rumors, moods, and everything imaginable. Most of the data on Twitter is available publicly and this data can be used to perform sentiment analysis or opinion mining which could prove extremely useful to companies launching consumer products.

Sentiment mining of people's tweets is a fast and efficient method to analyze the public opinion towards any topic of interest. This field has numerous applications which include customer feedback, prediction of elections, advertising and marketing, product or service reviews, movie box office and stock market predictions. The purpose of this paper is developing a system to perform the task of sentiment analysis, so as to summarize and analyze peoples' opinions about any particular event, product, service, person etc.

The rest of this paper has been organized as follows. The related work on sentiment analysis and short text classification has been discussed in Sect. 2. The architecture of the system proposed in this paper has been outlined in Sect. 3. The methodology and implementation details have been discussed in Sect. 4. The results have been presented and analyzed in Sect. 5, followed by the conclusions and future work Sect. 6.

2 Related Work

There has been quite a variety of research work done on opinion mining sentiment analysis over the years. The immense data available on social networking sites and these being popular venues for people to express their views has inspired researchers to experiment on sentiment analysis models for retrieving sentiment from such resources.

In [1] an approach is proposed wherein a publicized stream of tweets from the Twitter are gathered, preprocessed, and divided into positive, negative, and irrelevant classes based on their emotional content. This paper also analyzed and compares the performance of various classifying algorithms for sentiment analysis on Twitter data. This work was focused on the identification of acceptable classifiers based on their performance, which were used to divide tweets as polar, neutral, or irrelevant based on the expressed sentiment and then polar class is further subdivided into either positive or negative classes. In this study, the problems of opinion classification and sentiment analysis of Twitter data has been analyzed, which is very different from other opinion mining problems on detailed and structured messages.

A simple and elegant solution is provided in [2] for performing sentiment analysis on Twitter data. Firstly, the data was collected for the Twitter corpus as positive and negative tweets. Secondly, a simple sentiment classifier was built with the help of the common Naive Bayes classifier to determine the sentiment of a tweet. Thirdly, the classifier was tested against user tweets from five domains which are movies, news, jobs, sport, and finance. The results showed that it is quite feasible to use the Twitter corpus alone to classify a new tweet for certain domains. By using this approach, the time complexity of the system was greatly reduced. This paper proved that using the domain selection approach on the Naïve Bayes Algorithm to classify user sentiments was more efficient than the traditional approach.

A method in which a machine can automatically analyze the sentiment short text like tweets is proposed in [3]. This system was combined with manually labeled datasets of tweets to perform opinion mining. This system was built in such a way that the computer could automatically how to extract the tweets which contained polar opinions by filtering out the non-opinionated tweets followed by determining their sentiment class of the opinionated tweets as either positive or negative. In this paper, a system was designed which was a combination of supervised learning and opinion extraction to classify tweets.

A method to assess these identified types of emotions in a tweet using opinion mining is presented in [4]. A two-step approach is proposed, where firstly, to identify the sentiment, extract the opinion words (a combination of the adjectives along with the verbs and adverbs) in the tweets and subsequently use a novel algorithm to find the emotion values of opinion words. The initial results show that it is a motivating technique, which may find potential applications in business intelligence, government policy making, amongst others. The paper presents a method to assess fixed set of emotions (happiness, anger, sadness, fear and disgust) in a tweet or a set of tweets using opinion mining. The proposed framework firstly identifies the sentiment by extracting the opinion words (a combination of the adjectives along with the verbs and adverbs) in the tweets and subsequently makes use of a novel algorithm to find the emotion values of the opinion words. The overall value of the emotions in the set of tweets is then calculated using a linear equation.

The movie reviews were used as both training set and testing in [5]. This paper proposes an approach which combines adjective analysis and the naive Bayes classifier and classifying the sentiment of tweets. Firstly, naive Bayes was applied on the testing set which resulted in two sets of tweets (polarized tweets and ambiguous tweets). Adjective analysis is performed on the set of ambiguous tweets wherein they are divided into polar and non-polar tweets with the help of the polarity adverbs and adjectives. Naïve Bayes classifier is used to classify the polarized tweets into truly and falsely polarized tweets. The falsely polarized set was further processed by using adjective analysis to decide the tweet polarity. The polar tweets are finally divided into positive and negative classes. The efficiency of the system increased proportionally as more adverbs and adjectives were added to the corpus.

A unique solution to sentiment analysis of short and informal text is proposed in [6] with the focus on tweets. This paper focused on the processing of informal statements. The use of sentiment features such as emoji or emoticons has been widely prevalent in tweets. These sentiment features help in determining the sentiment of the text. Thus, a sentiment feature generator module was used to assign scores to sentiment features like emoticons. A higher weightage was given to these sentiment features in the calculation of the sentiment scores. A hybrid method was proposed in this paper to perform sentiment analysis of short text. This method was compared against the three most common classifiers of this field which are Maximum Entropy, Naïve Bayes and Support Vector Machines According to the experiments it was believed that with respect to sentiment analysis, using sentiment features rather than conventional text analysis yielded a greater accuracy.

From these works, it could be inferred that, the Naïve Bayes classifier is the most suitable machine learning algorithm to be used for the task of sentiment classification of short text. Most of these papers have made the use of tools such as WEKA (a data mining tool) for sentiment classification because their purpose has been to test the accuracies of different machine learning algorithms. On the other hand, the purpose of this paper is to build a generalized system to perform sentiment analysis by implementing an algorithm based on the Naïve Bayes classifier.

3 System Architecture

Firstly, tweets are collected from Twitter with respect to a query term and stored locally on the disk. The next step is to preprocess and clean the data to get the tweets in the proper format to sentiment analysis. Next, each tweet is tokenized (split) into meaningful words. After this, a custom Naïve Bayes algorithm is used to classify the tweets into positive, negative, and neutral tweets after which the results are analyzed and represented visually to get the summary of the overall opinion towards that topic. Figure 1 demonstrates the architecture of the system proposed in this paper.

4 Methodology

4.1 Data Collection

The first step was to stream proper data from Twitter which would be the test data. This was done by sending requests for data through a Twitter API called Twitter4 J by querying a search term or phrase. The number of tweets received depends on the Twitter data stream and is variable. But usually for popular search terms, at least a few thousand tweets are returned which are then stored locally on the disk. After receiving the data, the data is cleaned and pre-processed to provide a proper testing data set upon which sentiment analysis can be performed.

The training set for the Naïve Bayes classifier has been downloaded from Sentiment140 [7] which consists of manually labelled datasets of tweets as positive and negative. The training set was also pre-processed to increase the efficiency of the algorithm.

Test Set: thousands of tweets which are collected in real time through Twitter API Twitter4J.

Training Set: set of manually labelled positive and negative tweets downloaded from Sentiment140 [7]

4.2 Data Preprocessing

The data pre-processing steps include the following:

- Removal of tweets which are not in English
- Removal of usernames (marked by @ symbol)
- Removal of hashtags (marked by # symbol)
- Removal of hyperlinks (URLs)
- Removal of email ids
- Removal of stop words which are irrelevant in sentiment classification
- Removal of retweets symbol (marked by RT)
- Removal of numbers
- Removal of other unwanted special characters and unnecessary punctuation
- Compression of words (elimination of many repeated letters)

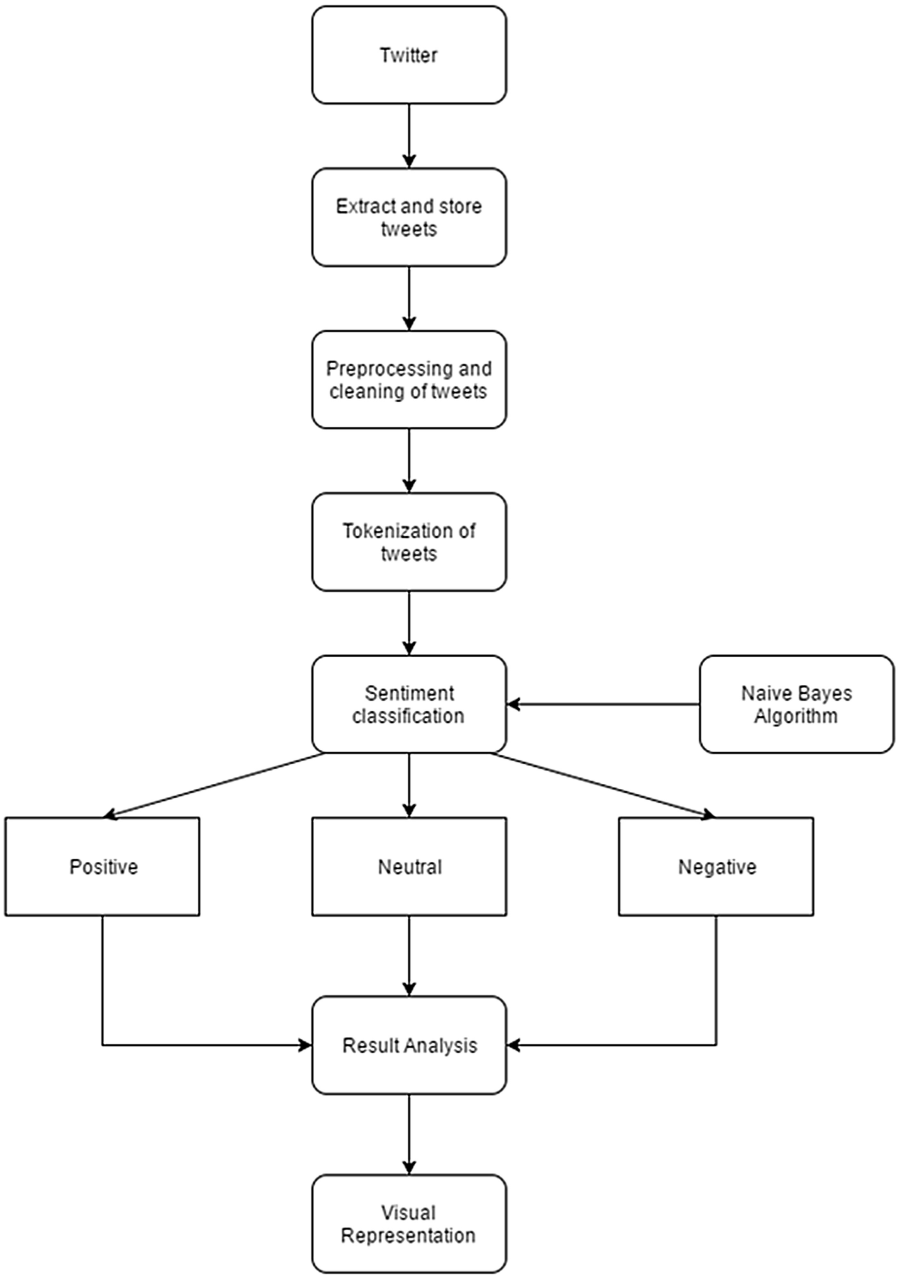


Fig. 1. System architecture

After cleaning the dataset into the proper format the next step is to tokenize (split) the tweet into separate words which is then used to perform sentiment analysis using a modified Naïve Bayes algorithm.

4.3 Algorithm for Performing Sentiment Analysis

A customized version of Naïve Bayes algorithm is used to determine the sentiment of the text. The Naive Bayes classifier uses a probabilistic model to decide which class best matches for a given input text. This classifier is called naive because it assumes independency between different features i.e. the value of a feature is independent of the value of any other feature for the given class.

Naïve Bayes classifiers belong to a family of probabilistic classifiers in the field of machine learning and are implemented by applying Bayes theorem with the assumption of independence between its features. The probabilistic model of the Naïve Bayes classifier is as follows:

With the help of Bayes Theorem, the conditional probability is calculated as given in Eq. (1)

$$p(C_k \mid \mathrm{x}) = \frac{p(C_k)\, p(\mathrm{x} \mid C_k)}{p(\mathrm{x})} \tag{1}$$

where $\mathrm{p}(C_\mathrm{k} \mid \mathrm{x})$ is the posterior probability to be calculated i.e. probability that instance x belongs to class $\mathrm{C_k}$, $\mathrm{p(C_k)}$ is the prior probability of class $\mathrm{C_k}$, $\mathrm{p(x \mid C_k)}$ is the likelihood of instance x belonging to class $\mathrm{C_k}$ and $\mathrm{p(x)}$ is the evidence i.e. the probability of instance x occuring.

In plain English using Bayesian terms, the previous equation can be rewritten as Eq. (2)

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{evidence}} \tag{2}$$

The naive Bayes algorithm uses a combination of this probabilistic model with a decision rule. The common rule is to select the hypothesis which has the most probability of occurring. The corresponding classifier, a Naïve Bayes classifier, is the function that assigns a class label $\hat{y} = C_k$ for some k, can be represented as Eq. (3)

$$\hat{y} = \underset{k \in \{1,\ldots,K\}}{\operatorname{argmax}}\ p(\mathrm{C}_k) \prod_{i=1}^{n} p(x_i \mid C_k). \tag{3}$$

Equation 3 denotes that the instance x is assigned the label $\hat{y}$ and belongs to the class $\mathrm{C_k}$ with the highest posterior probability among the calculated posterior probabilities of all classes.

This function is a common probabilistic Naïve Bayes classifier which has a variety of applications. In this paper it is being used for the task of sentiment analysis. The algorithm proposed in this paper is built upon this Naïve Bayes classifier with some modifications.

The unigram (bag of words) model is used in this algorithm. A modified version of the probabilistic model of the Naïve Bayes classifier is used to build this algorithm. The sentiment score for each tweet is calculated using the algorithm. This is done for all the tweets in the test data and the overall sentiment about that topic is analysed.

The modified algorithm used in this paper has been built around the Naïve Bayes classifier to classify the sentiment of short text such as tweets. A clause to perform negation handling has also been incorporated into this algorithm which is the major contribution of this paper to the Naïve Bayes algorithm.

Negation handling aims to increase the efficiency of the Naïve Bayes algorithm by correctly identifying the sentiment of double negatives or false positives. This is done by checking if the preceding word belongs to the set of negation words and thereby changing the sentiment score accordingly. The set of negation words consists of words such as no, not, never, can't, doesn't etc. For example, for the text "This is not bad" a normal Naïve Bayes classifier would assign it a negative sentiment score. But this modified algorithm recognizes "not bad" as a double negative and thus will assign the text a positive sentiment score. Similarly, the text "Rain is never good" would be normally identified as a positive sentiment but this algorithm would correctly identify it as a negative sentiment. Thus, negation handling increases the efficiency of sentiment classification.

The modified version of the Naïve Bayes Algorithm implemented is as follows:

Algorithm: Modified Naïve Bayes

Input: Test set (tweets collected in real time) and Training sets (C*pos*, C*neg* and NW)

Output: Sentiment

Symbolic representation meanings

```
n: number of tweets in test set
Cpos: set of tweets in the positive training file
Cneg: set of tweets in the negative training file
NW: set of negation words
Ti: tweet i in the test set (i=1... n)
P(Ti | Cpos): probability of tweet belonging to class Cpos
P(Ti | Cneg): probability of tweet belonging to class Cneg
m: number of words in tweet Ti
Wj: word j in the tweet Ti (j=1...m)
P(Wj): likelihood of word Wj occurring in class C
F(Wj): Frequency of word Wj in class C
P(Ti | C): probability of tweet belonging to class C
```

Algorithm

```
1: start
2: for each tweet Ti (i=1... n) in the test set do
3:  P (Ti | Cpos) ← Calculate (Ti, Cpos)
4:  P (Ti | Cneg) ← Calculate (Ti, Cneg)
5:  if P (Ti | Cpos) > P (Ti | Cneg) then
6:    Ti is positive
7:  end if
8:  if P(Ti | Cpos) < P(Ti | Cneg) then
9:    Ti is negative
10: end if
11: if P(Ti | Cpos) = P(Ti | Cneg) then
12:   Ti is neutral
13: end if
14: end for
15: end
```

Calculate function definition

```
1: function Calculate (Ti, C)
2: Prior ← (No. of tweets in C)/ (Total no. of tweets in the
training set)
3: Count ← Total no. of words in class C
4: P(Wj) ← 0
5: for each word Wj (j=1...m) in tweet Ti do
6:  F(Wj) ← No. of occurrences of word Wj in class C
7:  if Wj-1 ∈NW then
8:    P(Wj) ← P(Wj) - F(Wj) / Count
9:  else
10:   P(Wj) ← P(Wj) + F(Wj) / Count
11: end if
12: end for
13: P (Ti | C) ← Prior * P(Wj)
14: return P (Ti | C)
15: end function
```

Note: In some cases, P (Ti | Cpos) = P(Ti | Cneg) = 0 (because no words in Ti exist either in Cpos or Cneg) so Ti is classified as neutral or irrelevant.

The sentiment scores for all the tweets in the test set are calculated and compared thereby giving the sentiments for all the tweets in the dataset. The last step involves summarizing the sentiment scores of the classes (positive, negative, or neutral) by making use of proper data visualization tools.

5 Results

The Figs. 2, 3 and 4 show the outputs of the system and represent the overall sentiments towards the topics searched. Table 1 shows the overall sentiment about the entered keywords or phrases based on the user tweets collected from Twitter at the time of running the application. For the sake of clarity, the most polar results are depicted. From Table 1 it is evident that this system can be used to analyze the overall sentiment of any general topic, company, person etc. and not only domain specific search terms.

With the help of these results, companies, people, governments, customers, organizations, and other interested parties can make informed decisions before making their intended decision. This will also help keep things in perspective and give them an opportunity to analyze any missteps they have made, thereby allowing them to take the necessary steps to rectify or amend their mistakes. In any case, knowing the public opinion or sentiment is of utmost importance especially for elections and advertisement companies. Other applications can include product or service reviews, movie box office or stock market predictions. This is also a useful tool for companies to get customer feedback about their products.

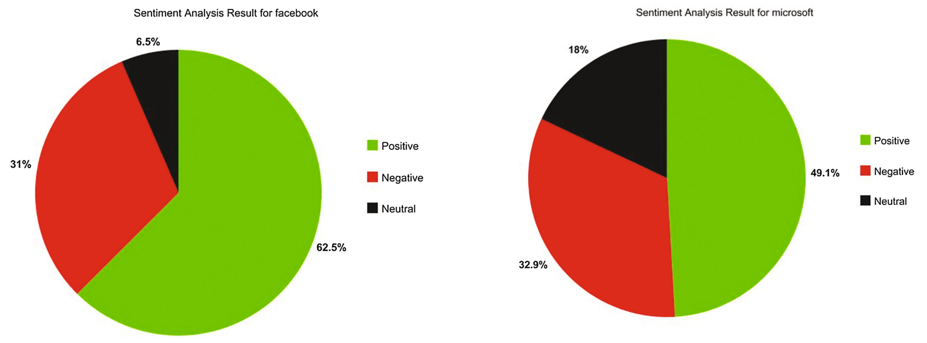


Fig. 2. Overall sentiment summary for the search term "facebook" and "microsoft"

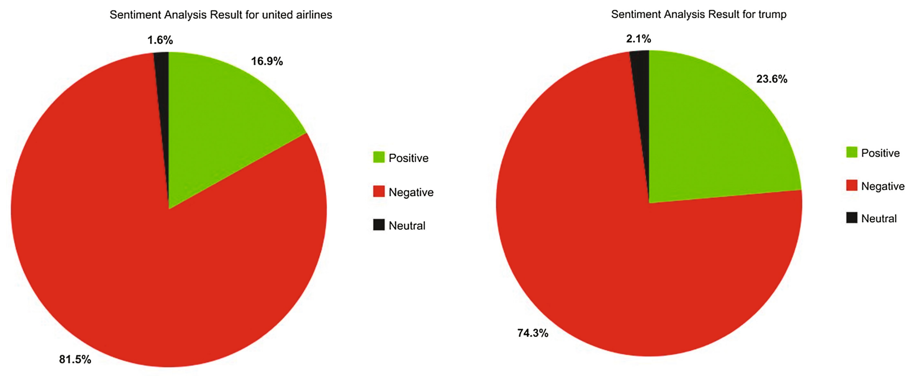


Fig. 3. Overall sentiment summary for the search term "united airlines" and "trump"

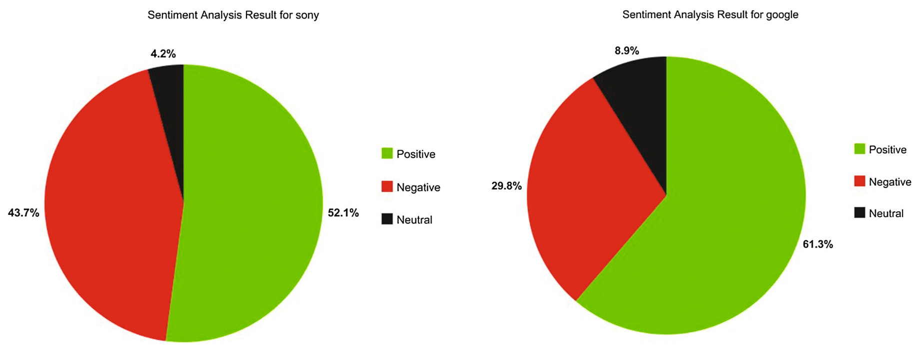


Fig. 4. Overall sentiment summary for the search term "sony" and "google"

Table 1. Overall sentiment

Search term	Overall sentiment (in %)		
	Positive	Neutral	Negative
Accenture	48.3	12.5	39.2
Airtel	37.1	6.6	56.3
Apple	48.1	7.4	44.5
Bsnl	39.7	10.6	49.7
Facebook	62.5	6.5	31.0
Google	61.3	8.9	29.8
Irctc	34.3	14.3	51.4
Jio	64.9	4.0	31.1
Micromax	22.5	19.1	58.4
Microsoft	49.1	18.0	32.9
Oneplus	53.0	7.3	39.7
Redmi	34.7	1.0	64.3
Samsung	61.3	2.2	36.6
Sony	52.1	4.2	43.7
Trump	23.6	2.1	74.3
United Airlines	16.9	1.6	81.5

6 Conclusions and Future Work

Social networking websites have become very popular and have attracted the attention of people worldwide to create social relations and collaboration in society. Opinion mining and sentiment analysis on Twitter data will only become more popular as time passes. Statistics prove that an increasing number of people are using this microblogging site to put forth their opinions. The huge amount of data contained in Twitter makes it a very attractive source of data for sentiment analysis. However, performing opinion mining or sentiment analysis on twitter data is a challenging task. The primary reason for this is that tweets include ambiguous words. People write in a different way than they intend to (words which are spelled differently but pronounced similarly or word with the same spelling but with different meanings). Here an algorithm based on the Naive Bayes classifier is designed to correctly predict the sentiment of short text such as tweets. This system uses a generalized training dataset to predict the sentiment of the tweets. There is a vast impact of this domain in many contemporary fields and a variety of applications are developed frequently in which sentiment analysis is used in many ways. Additional filters, modifications or tweaks can be added to this approach which include techniques to perform sarcasm detection, comparison handling, domain classification etc. To increase the accuracy of the system, the dataset can be made domain specific.

References

1. Priyanthan, P., Ragavan, T., Prasath, N., Perera, A.: Opinion mining and sentiment analysis on a twitter data stream. In: The International Conference on Advances in ICT for Emerging Regions, pp. 182–188 (2012)
2. Shahheidari, S., Dong, H., Bin Daud, M.N.R.: Twitter sentiment mining: a multi domain analysis. In: Seventh International Conference on Complex, Intelligent, and Software Intensive Systems, pp. 144–149 (2013)
3. Po-Wei, L., Bi-Ru, D.: Opinion mining on social media data. In: IEEE 14th International Conference on Mobile Data Management, vol. 2, pp. 91–96 (2013)
4. Kumar, A., Dogra, P., Dabas, V.: Emotion analysis of twitter using opinion mining. In: Eighth International Conference on Contemporary Computing (IC3), pp. 285–290 (2015)
5. Mertiya, M., Singh, A.: Combining Naive Bayes and adjective analysis for sentiment detection on Twitter. In: International Conference on Inventive Computation Technologies (ICICT), vol. 2, pp. 1–6 (2016)
6. Bahrainian, S.-A., Dengel, A.: Sentiment analysis using sentiment features. In: IEEE/WIC/ACM International Conferences on Web Intelligence (WI) and Intelligent Agent Technology (IAT), vol. 3, pp. 26–29 (2013)
7. Go, A., Bhayani, R., Huang, L.: For Academics - Sentiment 140 - A Twitter Sentiment Analysis Tool. http://help.sentiment140.com/for-students.html. Accesesed 29 Mar 2017

Cost-Sensitive Feature Selection for Class Imbalance Problem

Małgorzata Bach(✉) and Aleksandra Werner(✉)

Silesian University of Technology, Gliwice, Poland
{malgorzata.bach,aleksandra.werner}@polsl.pl

Abstract. The class imbalance problem is encountered in real-world applications of machine learning and results in suboptimal performance during data classification. This is especially true when data is not only imbalanced but also high dimensional. The class imbalance is very often accompanied by a high dimensionality of datasets and in such a case these problems should be considered together. Traditional feature selection methods usually assign the same weighting to samples from different classes when the samples are used to evaluate each feature. Therefore, they do not work good enough with imbalanced data. In situation when the costs of misclassification of different classes are diverse, cost-sensitive learning methods are often applied. These methods are usually used in the classification phase, but we propose to take the cost factors into consideration during the feature selection. In this study we analyse whether the use of cost-sensitive feature selection followed by resampling can give good results for mentioned problems. To evaluate tested methods three imbalanced and multidimensional datasets are considered and the performance of chosen feature selection methods and classifiers are analysed.

Keywords: Class imbalance problem · Feature selection · Cost sensitive learning · Classification

1 Introduction

One of the greatest challenges in machine learning and data mining research is the class imbalance problem which exists in real world applications. Examples of these kinds of applications include biological data analysis, text and image classification, web page classification, medical diagnosis/monitoring and many others. In most of the mentioned cases identifying rare objects is of crucial importance. It is more significant to classify correctly the minority class samples in comparison to the samples of the majority class, although the minority class is much smaller than the majority one. Unfortunately, the use of conventional learning methods for imbalanced data often results in constructing a decision model biased toward the majority class and consequently all objects are assigned to the dominant class.

Another problem in machine learning is high dimensionality of data. A deluge of information which can be observed nowadays is reflected in occurring the sets of data with hundreds or even thousands of features. High degree of irrelevant and redundant information which may greatly degrade the performance of learning algorithms, causes the suitable data manipulation is critical to cope with data overload. Moreover, the traditional feature selection methods do not consider the imbalance problem since they assign the same weighting to samples from different classes when these samples are used to evaluate the suitability of each feature.

Analyses show that in the case of strong data imbalance a lot of feature selection algorithms choose those features which are characteristic for the majority class. It worsens the classification accuracy of the minority cases. This is due to the fact that many of these methods are based on an instance,[1]. That is the ability to estimate the quality of features is highly dependent on the number of instances from different classes.

To our knowledge, the unambiguous guidelines how to deal with the problem of multidimensionality within the context of data imbalance do not exist. The problem is still open and needs more research efforts. Therefore, in the presented study we analyse whether considering the costs at the stage of feature selection can help in this issue.

The content of the paper is as follows. In the next section we discuss the matters concerning feature selection and class imbalance problem and give an overview of the related work. In Sect. 3 the proposed approach is outlined. There is also a short description of data used in our experiments as well as information about applied classifiers and their evaluation metrics. The results of the performed tests and the discussion of the outcomes are given in this part of the paper too. Finally, Sect. 4 presents the conclusions of the conducted research and suggests further research trends.

2 Background Knowledge

The problem related with imbalance of data is not new. Many machine learning approaches have been developed for the last decade to cope with imbalanced data classification. Despite the sharp rise in the number of publications in this area there are still many issues in this topic to deal with, particularly relating to learning from not only imbalanced, but also multidimensional datasets [1].

2.1 Data Imbalancing

As already mentioned, samples of the minority class are often more important than those from the majority one and their correct recognition is the main objective of the prediction. It means that bad recognition of the minority examples has much more serious consequences for the user than the creation of so-called ‘false alarm’, when the examples of the majority class are assigned to the minority one. It is particularly noticeable in identifying the rare and serious diseases. Unfortunately, typical search strategies optimize global criteria, such as error or entropy measures, which often result in constructing a decision model to the majority class.

[1] They are known as instance-based filtering methods.

A lot of various methods for handling class imbalance have been reported in the specialized literature [2]. One of the ways to remove skewed class distribution is sampling of dominant data and choosing a number of examples of each category which is too large in relation to others. This approach is called undersampling. Another method can be the replication of the examples of minority class and artificial 'reproducing' the examples of all categories with less cardinality. It is called oversampling.

The simplest preprocessing methods are both random undersampling and random oversampling. The major drawback of random undersampling is that it can discard potentially useful data which could be important for the learning process. On the other hand, for random oversampling the overfitting may occur more frequently because in this process the exact copies of existing instances are generated. To prevent this situation another solutions have been proposed. A good example is synthetic minority oversampling (SMOTE) [3], whose main idea is to form new minority class object by interpolating between several minority class examples which are adjacent. New instances are created by taking the existing example and its nearest neighbours. Accordingly, new examples combine features of the target case with features of its neighbours.

2.2 Data Multidimensionality

Dimensionality reduction may be achieved by feature selection or feature extraction. In the first approach variables with little impact on the result are rejected while in the second approach a new, low-dimensional vector of features is created, which consists of a combination of the input characteristics [4].

Feature selection methods, as a preprocessing step for learning algorithms, provide several benefits such as reduced computational costs, e.g. training time or storage requirements, and can also improve the performance of the prediction. Furthermore, reducing the number of features can improve significantly the understandability of the machine learning models and it often helps to build models with better generalization [5].

It seems that the best method for feature selection is to enumerate all the candidate subsets and then apply the appropriate measure to evaluate them. Unfortunately, the exhaustive search is often infeasible because the space of possible solutions is very large. An alternative way is the usage of a random search method where the candidate feature subset is generated randomly or the method called the heuristic search where the heuristic function is employed to search the optimal subset of features [5–7].

Existing feature selection techniques can be divided generally into three categories: filter, wrapper and embedded methods. They vary in a way of combining the selection algorithm and the model building:

- Filters select features regardless of the model. They are based only on general properties e.g. the correlation with the variable to predict. The fact that filters work without taking the classifier into consideration makes them very computationally efficient and more versatile [8].

- Wrappers use learning techniques to evaluate which features are useful. Evaluation of subsets of features is done by applying a classification model[2]. It means that the same model is used both to the selection and later to the classification, so feature subset found in the selection process is tailored to the particular algorithm.
- Embedded techniques merge the feature selection step and the classifier construction. The optimal set of features is built when the classifier is constructed, and the selection is affected by the hypotheses which are made by the classifier [8].

2.3 Feature Selection in Term of Data Imbalance

Good feature selection method should meet some criteria, namely: reliability, validity, reproducibility [9, 10]. Unfortunately, there are factors which can cause that these criteria are not fulfilled. One of them is a small number of samples in high-dimensional data.

It has been experimentally verified that the relatively small number of samples in high-dimensional data is one of the main sources of the instability problem in feature selection [11]. In [12] the authors have concluded that at least thousands of samples are needed to achieve stable feature selection.

Another source of instable results of feature selection is imbalance of the classes [13]. One of the ways to deal with the mentioned problem is to generate a number of artificial training samples. Then feature subsets can be assessed using both the generated data and the original ones. In our previous studies [14, 15] we tested some oversampling methods followed by some methods of feature selection. We have shown that such solution can have a positive impact on the classification accuracy. Nevertheless, various combinations of data balancing and feature selection methods are not always equally effective. This means that a lot of tests must be done to find possibly the most appropriate combination.

Currently, we analyse whether the inclusion of the costs at the stage of feature selection and performing it before oversampling with the use of SMOTE method can help to deal with the mentioned problem.

In our mentioned above papers the filters, wrappers as well as embedded methods were tested, and their advantages and disadvantages were discussed. In this study three feature ranking methods: Chi-square, Gain ratio and ReliefF were chosen.

1. Chi-square evaluates the worth of an attribute by computing the value of the chi-squared statistic with respect to the class. It is a two-sided metric, which measures the independence of the feature from the class labels based on the confusion matrix. This happens by assuming that there is a non-zero probability for an exact value to be drawn from the distribution, which leads to extremely small expected counts of feature values.
2. Information gain metric uses entropy to score the attribute relevance. It can be defined as the expected reduction in entropy caused by partitioning the examples according to a given attribute. It is biased towards attributes which have many values what may result in overfitting, i.e. in selecting the non-optimal set of attributes. Gain

[2] Selection is 'wrapped' around the model.

ratio is the modification of information gain that reduces its bias. It corrects the information gain by taking the intrinsic information of a split into account. This means that information about the class is ignored.
3. The ReliefF is an instance-based filtering method. It evaluates the worth of an attribute by repeatedly sampling an instance and considering the value of the given attribute for the nearest instance of the same and different class. For each feature in randomly selected sample, if that feature has a value different for that same feature for a nearest sample in the same class, then the algorithm takes this as not desirable and subtracts the distance from the weight of the attribute. If the value of the feature is different for a nearest sample in the different class, the algorithm decides this situation is desirable. It calculates the distance for that attribute and increases its weight.

Each of the above ranking methods (rankers) sorts features based on assigned worth and then chooses the best (first) and rejects the worst (last). The additional parameter, so called threshold, determines the cut. This may be a fixed number of features that should be left or the value of the index evaluation. In presented research we initially assumed to halve the number of features.

3 Cost-Sensitive Feature Selection - Experiments

The concept of the proposed approach is as follows.

1. Chosen methods of feature filtering are used for high dimensional datasets. The reduction is implemented using filter both with and without consideration of cost matrix. If the base evaluator can handle instance weights, then the training data is weighted according to the cost matrix, otherwise the training data is sampled according to this matrix.
2. For the datasets with the reduced vector of features the SMOTE method is introduced to oversample minority class.
3. The classification task is performed for data prepared in the above described way. In order to assess the performance of the filters with and without considering weighting some metrics, suitable for imbalanced data, are tested.
4. To make the analysis more complete, the obtained results are also compared with these reached on the basis of the original set of data (i.e. with full set of feature and without pre-selection of feature.

The choice of appropriate values for the cost matrix is a difficult task. Often, the weights are selected in such a way that the total cost of misclassification for each class is the same. This approach does not work in all cases and it is often necessary to choose other values which will better balance the negative effects of data skewness. In the presented research the higher costs are assigned for the misclassification of examples from the positive class with respect to the negative one. The cost for the minority class is calculated as the quotient of the number of samples in the majority class and the number of samples in the minority one.

It should be emphasized that both the feature selection and the oversampling were included inside the cross-validation procedure. The differences in classification accuracy are evaluated by Student t-paired test with 95% significance level. A general overview of the proposed concept is shown in Fig. 1.

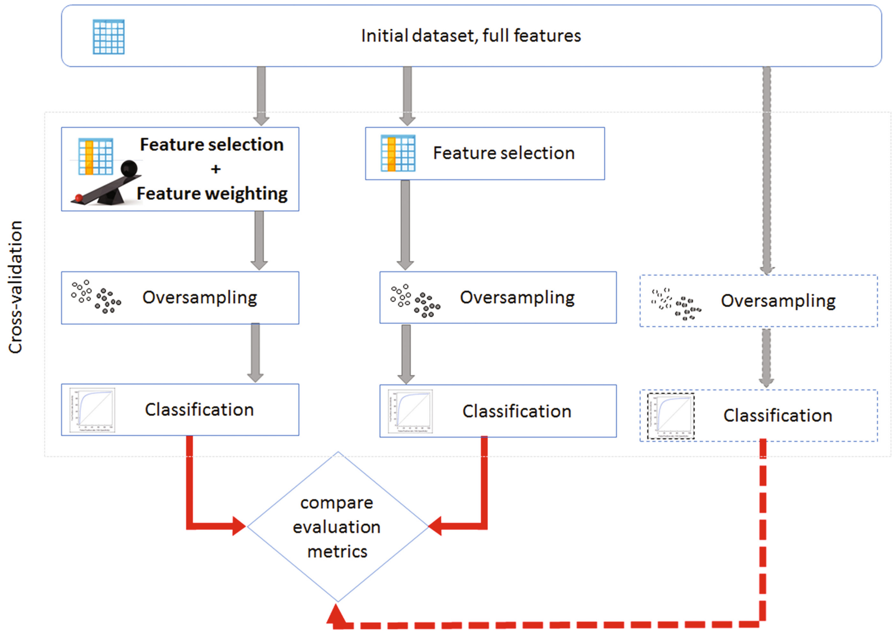


Fig. 1. The proposed scenario of the experiments

3.1 Experimental Setup

The research was performed using the Waikato Environment for Knowledge Analysis software Weka 3.7.13 [16] and R software environment [17]. Ten-fold cross-validation was used in the experiments and the final performance estimation was obtained from averaging of the results of the tenfold. It was stratified validation, which means that each fold contained roughly the same proportions of examples from each class.

For used feature selection methods and classifiers default values of parameters were applied, unless otherwise specified.

Data Sets
To evaluate tested methods three different datasets were used. An overview of the datasets is given in Table 1.

The first of listed sets was obtained from the epidemiological population-based study whose aim was to examine which parameters were important for prediction of osteoporotic fractures.

Table 1. Overview of datasets

Dataset	No. of samples	No. of features	%Minority class in the whole set
OSTEO	729	88	7.41
SECOM	1097	591	6.1
MADELON	1233	501	20

The second and third datasets were retrieved from the UCI Machine Learning Repository [18]. SECOM data was obtained by monitoring of a semi-conductor manufacturing process. It contains values of signals collected from sensors and other process measurement points.

MADELON is an artificial dataset which was the part of NIPS 2003 feature selection challenge. It was preliminary prepared by us, to simulate class imbalance problem.

Classifiers used in the tests

In the first phase of the research a set of the classifiers was defined to be used in the experiments. Following the Occam's Razor principle: use the least complicated algorithm that can address your needs and only go for something more complicated if strictly necessary' [19] and taking into account the classifiers which we used in the previous studies, for presented tests three algorithms of classification were chosen:

1. Naïve Bayes (NB). This classifier is based on the Bayes' theory which assumes independence between every pair of features in a given class and their equal contribution to the final model [20]. If an object is described by the values of *n* conditional attributes, the most probable class to which this object will be assigned is a class which maximizes the conditional probability.
2. K-nearest neighbours (KNN). The idea of KNN classifier is that similar examples belong to the same classes [21]. Hence, in order to assign a new example to a class, a certain number of its nearest neighbours is considered. In presented experiments one neighbour was selected for determining the output class. The LinearNNSearch with the Euclidean distance metric was used as the nearest neighbour search algorithm.
3. C4.5 decision tree[3]. It is one of the most widely used learning algorithms. It is the successor to ID3 developed in 1986 by Ross Quinlan [22] and it converts the trained trees, i.e. the output of the ID3 algorithm, into sets of if-then rules. The accuracy of each rule is then evaluated to determine the order in which the rules should be applied. In the next step pruning of too big branches is done by removing a rule's precondition if the accuracy of the rule improves without it. In the tests performed for C4.5 classifier the confidence level was set to 0.25 and the minimum number of item-sets per leaf to 2.

Evaluation metrics for data classification

A lot of metrics which allow to assess the performance of a classification can be found in the subject literature, but for the imbalanced data only some of them are especially essential [23]. In presented study the following metrics were analysed: specificity,

[3] In Weka data mining tool the implementation of the C4.5 algorithm is called J48.

sensitivity, geometric mean (GMean), balanced accuracy (BAcc) and Receiver Operator Characteristic (ROC).

Sensitivity can be regarded as the percentage of positive cases/instances correctly classified as belonging to the positive class, while specificity as the percentage of negative cases correctly classified as belonging to the negative class. GMean, for binary classification, is the squared root of the product of the sensitivity and specificity. Balanced accuracy is the arithmetic mean of the sensitivity and the specificity. ROC chart visualizes the trade-off between the benefits (Sensitivity) and costs (1-Specificity). The area under curve (AUC) is calculated for the ROC chart, using the trapezoid method.

3.2 Results and Discussion

To make our experiments more intelligible their description is divided into some stages. For each of them, the research goals and scenarios are first presented, and then the tests results are discussed.

Stage #1

Three methods of filtering described in point 2.3 were used for each of the tested datasets which were presented in point 3.1. Each method was applied both for filters with the use of cost matrix[4] and without.

Thanks to this 2*(3*3) = 18 datasets with the reduced feature vectors were obtained.

The SMOTE method was introduced to oversample minority class for these datasets and 3 original ones. Each minority class was oversampled at 100% and 200%[5] of its original size. This resulted in 3*21 = 63 datasets for further tests.

Stage #2

For all above mentioned datasets three classifiers – Naïve Bayes, KNN and C4.5 were tested in this step. Ten-fold stratified cross validation was used. Each test was repeated 10 times and obtained values were averaged.

Summarizing, this gave a total of 3*63*100 = 18900 runs of the classification tasks. Each time 5 metrics: sensitivity, specificity, BAcc, GMean and AUC were evaluated.

The results achieved for selected classifier and filter are presented in Table 2[6].

These results concern the SECOM dataset, in respect of which C4.5 classifier and Chi-Squared filter were used.

One can see that employing Chi-Squared filter which assigns higher weight for minority class gives better results than obtained for the original sets of feature. The observed differences are statistically significant. The comparison with the outcomes for the filter without considering the costs is not so clear. For example, according to the AUC metric, the version without weighting is the best for SMOTE 200 dataset.

[4] In the presented research the cost for the minority class was calculated as the quotient of number of samples in the majority and minority classes.

[5] In the following paragraphs it is denoted as SMOTE 100 and SMOTE 200.

[6] The standard deviation was determined for all of the analysed metrics, but unfortunately due to volume limitations it is presented only for AUC.

According to Sensitivity, BAcc or GMean metrics the cost-sensitive version of this filter is better.

Table 2. Performance measures for C4.5 classifier and Chi-Squared filter

Balance	Set of features[a]	Sensitivity	Specificity	BAcc	GMean	AUC
Original	Original	0.09	0.962	0.526	0.294	0.57 ± 0.12
	Chi-Squared	0.06	0.95	0.505	0.239	0.57 ± 0.07
	Chi-Squared_weighted	0.119	0.968	0.543	0.339	0.661 ± 0.12
SMOTE 100	Original	0.463	0.937	0.7	0.659	0.69 ± 0.12
	Chi-Squared	0.455	0.935	0.695	0.652	0.71 ± 0.08
	Chi-Squared_weighted	0.56	0.949	0.755	0.729	0.73 ± 0.07
SMOTE 200	Original	0.602	0.941	0.772	0.753	0.76 ± 0.09
	Chi-Squared	0.632	0.926	0.778	0.765	0.81 ± 0.05
	Chi-Squared_weighted	0.637	0.92	0.779	0.766	0.79 ± 0.05

[a] *Original* denotes full set of features – i.e. without pre-selection of features. *Chi-Squared* and *Chi-Squared_weighted* denote reduced vectors of features. The reduction was implemented using filter both not considering and considering cost matrix, respectively.

Raeder et al. in [25] draw attention to the fact that the choice of the evaluation metric can affect the assessment of which tested methods are considered to be the best. Our results confirm that the various combinations of the classifiers and filters can be ranked differently by various evaluation measures.

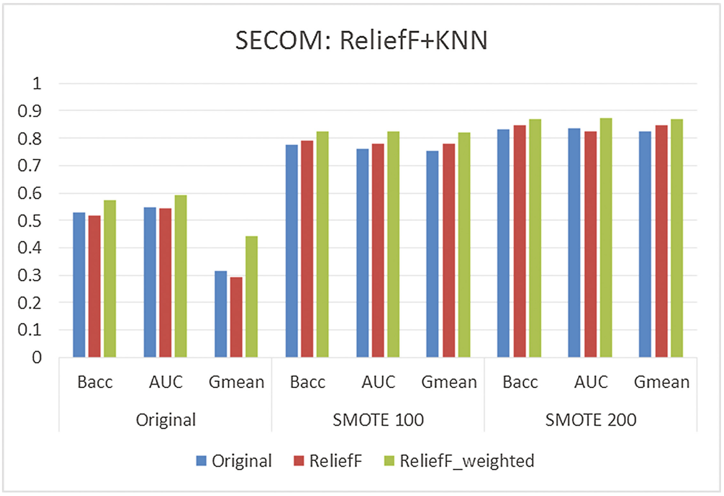


Fig. 2. Evaluation of SECOM dataset classification

This fact is additionally illustrated for BAcc, AUC and GMean metrics in Fig. 2. There are presented the outcomes for combinations: SECOM dataset, ReliefF filter and KNN classifier. In this case the use of the filter which considers cost matrix gives the best results for all presented metrics. However, the granting of the second place in the ranking depends on the analysed evaluation measure.

The outcomes for OSTEO dataset, for which Naive Bayes classifier and Chi-Squared filter were used, are presented in Fig. 3. The chart of ROC curves for mentioned

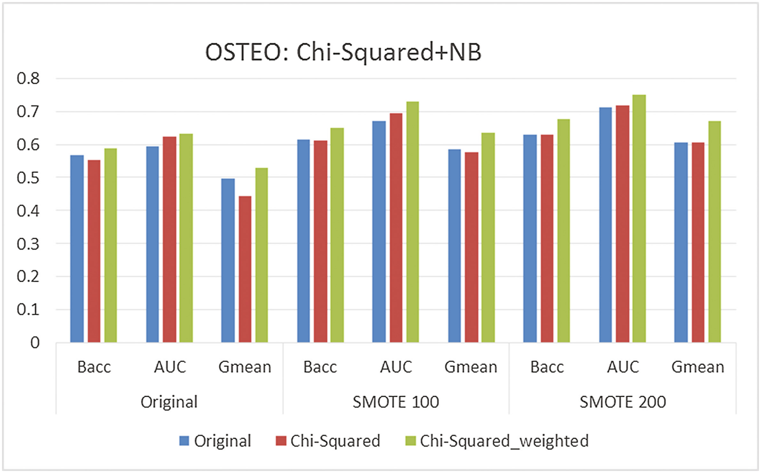


Fig. 3. Evaluation of OSTEO dataset classification

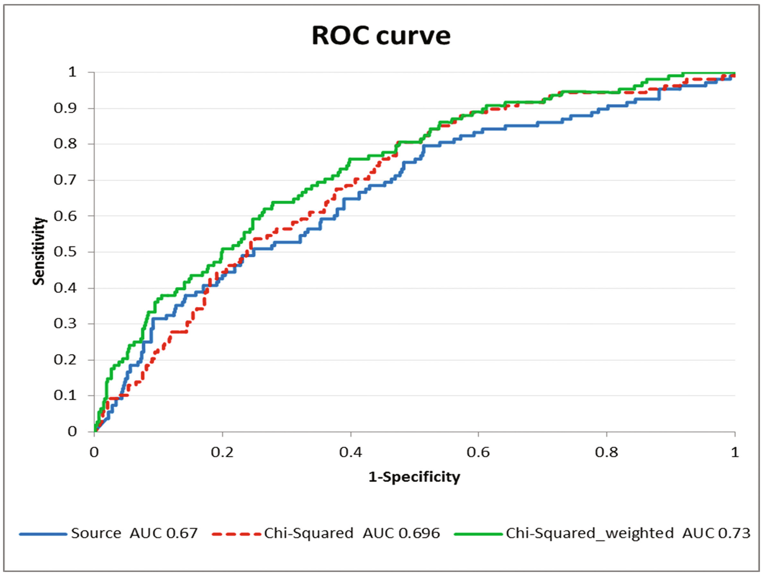


Fig. 4. ROC curve for Naïve Bayes algorithm and OSTEO dataset

combination of the classifier and the filter and for the minority class oversampled at 100% of its original size is additionally given in Fig. 4.

It can be noted that using Chi-Squared filter and taking into account the cost matrix give good results in this case. The application of cost-sensitive Chi-Squared filter before oversampling with SMOTE method allowed to obtain AUC = 0.73 ± 0.08. It is better result than that attained for the version without the prior feature selection (AUC = 0.67 ± 0.08) as well as compared to the feature selection in the variant without using cost matrix (AUC = 0.696 ± 0.07). Observed differences are statistically significant.

Stage #3

Each of the analysed classifiers was assessed in terms of its cooperation with different filters. In this phase we created filters' rankings for each of 3 analysed classifiers and separately for three balancing levels of each dataset (3*3). This gave a total of 3*3*3 = 27 rankings. The set of ranking positions obtained for each analysed dataset was the basis for the average values' calculation for each classifier. The number one in the ranking was assigned to the filter which allowed to achieve the greatest value of AUC for considered classifier[7].

It was concluded that C4.5 classifier works the best with the GainRatio filter (first ranking position). This can be explained by the fact that this classifier uses the gain ratio as the splitting criterion during the construction of decision trees. For KNN classifier the best results were obtained in a juxtaposition with the outcomes of the ReliefF filter. On the other hand, for Naive Bayes comparable results were achieved for the Chi-Squared and GainRatio filters. In this case a combination with the ReliefF filter gave the worst results.

It should be also noted that the results depend strongly on the characteristic of a dataset. Some differences in positioning of particular filters in rankings created for various datasets were observed. For example the Chi-Squared test works well for nominal values, but significantly worse for continuous ones. The use of the Chi-Squared_weighted filter and then the Naive Bayes classifier for OSTEO dataset resulted in a 4–10% increase of classification accuracy depending on the evaluation measure. The application of the same combination of the filter and classifier for SECOM and MADELON datasets gave significantly worse results. For some metrics even worse than those obtained for the original set of features. This was due to the fact that all the features in these datasets were of continuous type while in OSTEO some of the variables had the discrete values.

[7] Due to volume limitations we do not present all of the rankings, but only discuss their results in the text.

4 Summary

One of the main objectives of reducing the dimensionality of the features is to decrease the cost of their extraction and storage. This is especially important in the case of imbalanced data, where the majority of the cases belong to the less significant one. The reduction of computational complexity and execution time of classification is also not insignificant.

Moreover, reducing the number of features is sometimes necessary to make the machine learning model more understandable. Unfortunately, the feature vector reduction can sometimes cause a decrease of the classification accuracy, especially in the case of imbalanced data. The results presented in this paper show that applying the costs at the stage of feature selection may minimize any loss of accuracy or even improve it. However, this requires the thorough analysis of both the characteristic of tested data and the possibility to use different combinations of feature filters and classifiers in respect of the data.

The presence/absence of noisy or strongly correlated variables is a prerequisite for the success of any feature selection algorithm, but benefits achieved with feature selection also depend on the characteristics of the dataset. That is, inter alia, due to the fact that various feature selection algorithms promote various types of features. As shown, some algorithms are better at coping with continuous variables, while others prefer discrete ones. Therefore, the appropriate selection of the features selection method to the nature of the data is necessary.

In presented research the fixed number of features chosen by tested feature selection methods was assumed. Some researchers point out that proper setting of parameters can significantly improve the final result of classification task [24]. Therefore, to explore the presented problem better, each ranking method should be tested for various number of attributes. Selection of the optimal feature subset is a separate research problem and it will be the subject of further analyses. Besides, we plan to consider other cost functions, because the optimization of weighted cost may bring the superior performance of imbalanced multidimensional data classification.

Acknowledgements. Project financed from the Polish funds for learning in 2017 year.

References

1. Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., Bing, G.: Learning from class-imbalanced data. Expert Syst. Appl. **73**, 220–239 (2016). doi:10.1016/j.eswa.2016.12.035. Elsevier
2. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE TKDE **21**(9), 1263–1284 (2009)
3. Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W.: SMOTE: synthetic minority over-sampling technique. AIR J. **16**, 321–357 (2002)
4. Motoda, H., Liu, H.: Feature selection, extraction and construction. Commun. IICM **5**, 67–72 (2012)
5. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. **3**, 1157–1182 (2003)

6. Kononenko, I.: Estimating attributes: analysis and extension of relief. In: Proceedings of European Conference on Machine Learning, pp. 171–182 (1994)
7. Saeys, Y., Inza, I., Larrañaga, P.: A review of feature selection techniques in bioinformatics. Bioinformatics **23**(19), 2507–2517 (2007)
8. Hira, Z.M., Gillies, D.F.: A review of feature selection and feature extraction methods applied on microarray data. Adv. Bioinf. **2015**, 1–13 (2015). doi:10.1155/2015/198363
9. Neumann, U., Riemenschneider, M., Sowa, J.P., Baars, T., Kälsch, J., Canbay, A., Heider, D.: Compensation of feature selection biases accompanied with improved predictive performance for binary classification by using a novel ensemble feature selection approach. BioData Min. **9**, 36 (2016). doi:10.1186/s13040-016-0114-4
10. He, Z., Yu, W.: Stable feature selection for biomarker discovery. Comput. Biol. Chem. **34**, 215–225 (2010). Elsevier
11. Loscalzo, L., Yu, C.D.: Consensus group stable feature selection. In: Proceeding of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 567–575 (2009)
12. Ein-Dor, L., Zuk, O., Domany, E.: Thousands of samples are needed to generate a robust gene list for predicting outcome in cancer. Proc. Nat. Acad. Sci. U.S.A. **103**(15), 5923–5928 (2006)
13. Yang, P., Liu, W., Zhou, B.B, Chawla, S., Zomaya, A.: Ensemble- based wrapper methods for feature selection and class imbalance learning. In: PAKDD, Advances in Knowledge Discovery and Data Mining. LNCS, vol. 7818, pp. 544–555 (2013)
14. Werner, A., Bach, M., Pluskiewicz, W.: The study of preprocessing methods' utility in analysis of multidimensional and highly imbalanced medical data. In: Proceedings of 11th International Conference Internet in the Information Society 2016, pp. 71–87 (2016). ISBN: 978-83-65621-00-9
15. Bach, M., Werner, A., Żywiec, J., Pluskiewicz, W.: The study of under- and over-sampling methods' utility in analysis of highly imbalanced data on osteoporosis. Inf. Sci. **384**, 174–190 (2016). doi:10.1016/j.ins.2016.09.038. Si: Life Sci. Data Analysis. Elsevier
16. WEKA download page. http://www.cs.waikato.ac.nz/ml/weka/down-loading.html. Last accessed 10 Apr 2017
17. The R Project for Statistical Computing, web page. https://www.r-project.org/. Last accessed 10 Apr 2017
18. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml/index.html
19. Ashari, A., Paryudi, I., et al.: Performance comparison between Naïve Bayes, decision tree and k-nearest neighbor in searching alternative design in an energy simulation tool. Int. J. Adv. Comput. Sci. Appl. (IJACSA) **4**, 33–39 (2013). doi:10.14569/IJACSA.2013.041105
20. John, G.H., Langley, P.: Estimating continuous distributions in Bayesian classifiers. In: Eleventh Conference on Uncertainty in Artificial Intelligence, pp. 338–345 (1995)
21. Aha, D., Kibler, D.: Instance-based learning algorithms. Mach. Learn. **6**, 37–66 (1991)
22. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Francisco (1993). ISBN: 1-55860-238-0
23. López, V., Fernandez, A., Garcia, S., Palade, V., Herrera, F.: An insight into classification with imbalanced data: empirical results and current trends on using data intrinsic characteristics. Inf. Sci. **250**, 113–141 (2013). doi:10.1016/j.ins.2013.07.007
24. Kostrzewa, D., Brzeski, R.: Parametric optimization of the selected classifiers in binary classification. In: Advanced Topics in Intelligent Information and Database Systems, pp. 59–69 (2017). doi:10.1007/978-3-319-56660-3_6
25. Raeder, T., Forman, G., Chawla, N.V.: Learning from imbalanced data: evaluation matters, ISRL 23. In: Holmes, D.E., Jain, L.C. (eds.) Data Mining: Foundations & Intelligent Paradigms, pp. 315–331. Springer-Verlag (2012)

Constraint-Based Method for Mining Colossal Patterns in High Dimensional Databases

Thanh-Long Nguyen[1,2], Bay Vo[3(✉)], Bao Huynh[2,4], Vaclav Snasel[2], and Loan T.T. Nguyen[5,6]

[1] Center for Information Technology, Ho Chi Minh City University of Food Industry, Ho Chi Minh City, Vietnam
longthng@gmail.com

[2] VŠB-Technical University of Ostrava, Ostrava-Poruba, Czech Republic
huynhquocbao@tdt.edu.vn, vaclav.snasel@vsb.cz

[3] Faculty of Information Technology, Ho Chi Minh City University of Technology, Ho Chi Minh City, Vietnam
bayvodinh@gmail.com

[4] Center for Applied Information Technology, Ton Duc Thang University, Ho Chi Minh City, Vietnam

[5] Faculty of Information Technology, Nguyen Tat Thanh University, Ho Chi Minh City, Vietnam
nthithuyloan@gmail.com

[6] Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, Warsaw, Poland

Abstract. Constraint-based methods for mining patterns have been developed in recent years. They are based on top-down manner to prune candidate patterns. However, for colossal pattern mining, bottom-up manners are efficient methods, so the previous approaches for pruning candidate patterns based on top-down manner cannot apply to colossal pattern mining with constraint when using bottom-up manner. In this paper, we state the problem of mining colossal pattern with pattern constraints. Next, we develop a theorem for efficient pruning candidate patterns with bottom-up manner. Finally, we propose an efficient algorithm for mining colossal patterns with pattern constraints based on this theorem.

Keywords: Bottom up · Colossal patterns · Data mining · Itemset constraint · High dimensional databases

1 Introduction

Frequent pattern mining (FPM) has been a vital subject in the field of data mining first introduced by Agarwal [1] and has been a focused theme in research for over three previous decades. Frequent patterns (FPs) are referred to itemsets, subsequences, or substructures whose appearance frequency in the target dataset is not less than a user-specified threshold value. FPM plays a fundamental role in association rule mining [33, 34], correlation mining [4], sequential pattern mining [15, 24], episode mining [3, 35], partial periodicity [14], classification [13], clustering [6], etc. FPM has multiple applications such as market basket analysis [21], weblog analysis [29, 30, 37], DNA sequence analysis [22] and prediction [20].

Although numerous efficient algorithms have been proposed to solve frequent pattern mining or some of its variants, and the interest in this problem still persists [11], especially for FPM on huge database.

One of the main reasons for the high level of interest in FPM algorithms is due to the computational challenge of the task. The search space of FPM is enormous, which is exponential to the length of the transactions in the dataset. This is challenges for itemset generation when the support levels are low.

In 2007, Zhu et al. [36] proposed Pattern-Fusion algorithm for mining colossal patterns. This algorithm overcomes the huge search space by only mining *K* good frequent patterns. Dabbiru and Shashi [8] proposed the CMP (Colossal Pattern Miner) algorithm for mining colossal patterns. Next, Sohrabi and Barforoush [25] proposed a method named BVBUC for mining colossal patterns in high dimensional databases which uses a bottom-up strategy based on transactions. The BVBUC joins 1-transactions together to generate 2-transactions, and so on, until the number of transactions reaches minimum support threshold (*minSup*). Moreover, the authors also proposed a formula to prune branches that cannot expand to reach *minSup* to reduce the search space. Although BVBUC is faster than CMP and Pattern-Fusion, it has some limitations. Therefore, Nguyen et al. [17] proposed two algorithms, named CP-Miner and PCP-Miner, for efficient mining colossal patterns. CP-Miner is based on CP-tree to efficient mine colossal patterns while PCP-Miner is based on CP-Miner and pruning strategies to efficient prune nodes in CP-tree.

Although there are a lot of algorithms developed to solve the problem of mining colossal patterns, no algorithm is developed for mining colossal patterns with constraints. In fact, the users can put their constraint to mine the patterns that they concern. For example, they may want to mine patterns that contain an item in a set of items or patterns that contain a pattern.

For pattern constraints, there are many contributions related to mine frequent patterns with pattern constraints [5, 7, 9, 12, 16, 23, 26], mine class association rules with itemset constraint [18] and classes constraint [19], mine erasable itemsets with subset and superset constraints [28]. However, all of them are based on top-down manner to prune candidate patterns. Therefore, proposed strategies in these works cannot apply to colossal pattern mining with pattern constraint using bottom-up manner.

In this paper, we propose a method for mining colossal patterns with pattern constraint. The main contributions of this paper are as follows:

1. We state the problem of mining colossal patterns with pattern constraint.
2. A theorem is developed to prune nodes in CP-tree.
3. Based on the theorem, we propose an algorithm for fast mining colossal patterns with pattern constraint.

The rest of this paper is organized as follows. Section 2 presents some works related to FPM and FPM with constraints and mining colossal patterns. Section 3 presents basic concepts and problem statement. Section 4 presents a new theorem for fast pruning candidates. An efficient algorithm for mining colossal patterns with pattern constraint is also proposed in this section. In Sect. 5 we compare the proposed algorithms with post processing of PCP-Miner. Finally, Sect. 6 gives our conclusions and some future research directions.

2 Related Works

2.1 Mining Colossal Patterns

The concept of colossal patterns was developed to solve the problem of mining patterns in high dimensional databases [36]. Pattern-Fusion algorithm was proposed to mine K best patterns from the database with the number of items is large. A heuristic-based approach to approximate mine colossal patterns is also developed in Pattern-Fusion. To mine all colossal patterns, Sohrabi and Barforoush [25] proposed the BVBUC algorithm, which uses bit vectors to present the pattern in each transaction, and uses vertical bottom-up traversing in transactions to mine colossal patterns. BVBUC only mine patterns that their support is equal to *minSup*, the other patterns that support greater than *minSup* are pruned because their patterns are the subsets of the mined patterns.

Although BVBUC can solve the problem of mining all colossal patterns by using a bottom-up manner, which is very efficient when *minSup* is small. The search space of BVBUC increases very fast when the number of transactions or *minSup* increases.

Therefore, Nguyen et al. [28] developed CP (colossal pattern)-tree and CP-Miner algorithm for efficient mining colossal patterns. CP-Miner also uses bottom-up manner like BVBUC. It, however, uses efficient pruning techniques (such as (i) early pruning transactions; (ii) fast computing the pattern of a set of transactions; (iii) early pruning nodes that cannot be colossal patterns) to reduce the runtime and memory usage. PCP-Miner, an improved version of CP-Miner, was also developed by Nguyen et al. [28]. Like CP-Miner, PCP-Miner uses CP-tree to mine patterns. A sorting strategy is developed to efficient prune candidate patterns and therefore, the runtime and memory usage of PCP-Miner reduce significant compared to CP-Miner.

2.2 Mining Pattern with Constraints

There are three approaches for mining FPs with constraints: The first approach is called post-processing method. They are based on Apriori [2], Eclat [31, 32] or FP-Growth [10] to mine all FPs and then filter out the FPs satisfy the constraint. This approach generates many candidates, meaning that it is time-consuming. In the real world, the users are interesting in a small set of FPs which satisfy conditions. The second approach is called pre-processing. In this approach, firstly, they discarded all the records do not satisfy the constraint. Then, based on the rest of database they mine all FPs. For example, dataset filtering [30] and MCFPTree [12]. And, the last approach is called constraint FPs. The authors integrated constraint into the mining process so that it was generate all the FPs which satisfy the constraint. Example CAP [16] and MINE_FS_CONS [27]. This approach can mine enough the constraint FPs for end users.

3 Basic Concepts and Problem Statement

Definition 1 (Support of a pattern) [36]. Given a transaction dataset *D*, the support of a pattern *X*, denoted by sup(*X*), is the number of transactions that contain *X*.

Definition 2 (Frequent pattern) [36]. Given a transaction dataset *D*, a pattern *X* is frequent if sup(*X*) $\geq$ *minSup*.

Definition 3 (Core Pattern) [36]. Given a pattern *Y*, a pattern $X \subseteq Y$ is said to be a τ-core pattern of *Y* if sup(*Y*)/sup(*X*) $\geq \tau$, $0 < \tau \leq 1$ (τ is called the core ratio).

For a pattern *Y*, let *C–Y* be the set of all its core patterns, i.e., $CY = \{X \mid X \subseteq Y$ and X is a τ-core pattern of $Y\}$.

Definition 4 (Colossal pattern) [36]. FP is a set of all frequent patterns in a transaction database *D*. A pattern *X* is called a colossal pattern in FP if and only if there does not exist an itemset *Y* such that $X \subset Y$ and *X* is a τ-core pattern of *Y*.

The problem of mining colossal patterns is to find colossal patterns that satisfy the core ratio τ.

Based on the problem of mining colossal patterns, we state the problem of mining colossal patterns with itemset constraint as follow:

Definition 5 (Colossal pattern with pattern constraint). Given a transaction dataset *D*, a core ratio τ, and a pattern *X*, mining colossal patterns with pattern constraint is to mine all colossal patterns in *D* that their patterns contains *X*.

For example: Consider an example database shown in Table 1 with $\tau = 0.375$ and *X* = DE, we have colossal patterns with *X*-constraint as shown in Table 2.

Table 1. An example database

Tid	Items
1	B, E, F, H, G, K
2	B, C, D, E, F
3	A, D, E, F, L
4	G, K
5	A, D, E, J
6	H, K, L
7	A, B, C, D, E, G, K
8	A, B, C, D, E, F, H

Table 2. Colossal patterns with DE-constraint

#	Colossal pattern
1	CDE
2	DEF
3	CDEF
4	ADEF

4 Proposed Algorithm

In this section, we develop a theorem to fast prune candidate patterns that cannot satisfy pattern constraint. After that, we propose an efficient algorithm for mining colossal patterns with pattern constraint.

Definition 6 (Subsuming of a pattern). Given two patterns P1 and P2, if P1 $\subseteq$ P2 then P1 is subsumed by P2.

Theorem 1 [28]. If pattern X of a node is subsumed by any colossal pattern, then all patterns created from this node cannot be colossal patterns.

Theorem 2. Given a node n in the CP-tree, if n.pattern does not satisfy pattern constraint then all its child nodes cannot satisfy pattern constraint.

Proof. According to Definition 5 and the hypothesis, n.pattern does not contain X, i.e. n.pattern $\cap X \subset X$. Besides, Y is a child node of n then $Y \subset n$.pattern. It leads to $Y \cap X \subset X$ or Y cannot satisfy pattern constraint.

For example: Consider X = CDE from the database in Table 1. When we join two nodes 1 $\times$ BEFHGK and 2 $\times$ BCDEF into node 12 $\times$ BEF, because BEF does not satisfy X (i.e., BEF does not contain X) $\Rightarrow$ all child nodes of 12 $\times$ BEF do not satisfy X-constraint.

Figure 1 presents the CPCP-Miner algorithm for mining colossal patterns that satisfy the core ratio τ and pattern X. Basically, it is based on PCP-Miner [28]. Main different of these two algorithms is in Lines 2 and 15. In these lines, CPCP-Miner is based on Theorem 2 to prune nodes early.

```
Input: Transaction database $D$, core ratio $\tau$, and pattern constraint $X$.
Output: Colossal patterns $CP$ that satisfy $X$.
Method:
  1.  $minSup = \lceil \tau \times |D| \rceil$;
  2.  $D' = D$ after filtering items that do not satisfy $minSup$, removing zero transactions
      and removing items that cannot satisfy $X$; // by Theorem 2
  3.  $[\varnothing] = \{\{t_1,1\}, \{t_2,1\}, ..., \{t_m,1\}\}$ from $D'$;
  4.  $CP = \varnothing$;
  5.  CPCP-Miner($[\varnothing]$, $CP$, $minSup$);

Procedure CPCP-Miner($[T]$, $CP$, $minSup$)
  1.  Sort-Pattern($[T]$);
  2.  If $T$.sup = $minSup$-1 then
  3.      For all $n$ in $[T]$ do
  4.          If Checking-Colossal($n$.pattern) then
  5.              Insert $n$.pattern and $n$.sup to $CP$;
  6.  Else
  7.      For all $n_i$ in $[T]$ do
  8.          If ($n_i$.sup + $|[T]|$ - i $\geq$ minSup) then
  9.              $[T1] = \varnothing$;
  10.             For all $n_j$ in $[T]$ with j > i do
  11.                 $Y$.pattern = $n_i$.pattern $\cap$ $n_j$.pattern;
  12.                 Y.sup = $n_i$.sup +1;
  13.                 If ($|Y$.pattern$|$ = $|$ $n_j$.pattern$|$) then
  14.                     Remove $n_j$ from $[T]$; // by Theorem 1
  15.                 If ($X \subseteq Y$.pattern) then // by Theorem 2
  16.                     Add $Y$ into $[T1]$;
  17.             CPCP-Miner($[T1]$, $CP$, $minSup$);
```

Fig. 1. CPCP-Miner algorithm for mining colossal patterns with pattern constraint

5 Experiments

5.1 Experimental Environment and Databases

All experiments presented in this section were performed on a PC with an Intel Core i5 3.2 GHz, and 4 GB of Ram, running on Windows 7 operating system, with the Visual C# 2010.

Table 3. Characteristics of the experimental databases

Database	# of items	# of transactions
Accidents	468	340,183
Connect	130	67,557
Retails	16,470	88,162
Pumsb*	7,117	49,046
T10I4D100 K	1,000	100,000

We made experiments on five databases, as shown in Table 3, which were downloaded from http://fimi.cs.helsinki.fi/data/.

5.2 Comparison of the Runtimes

Figures 2, 3, 4 and 5 show the runtimes for PCP-Miner with post processing and CPCP-Miner.

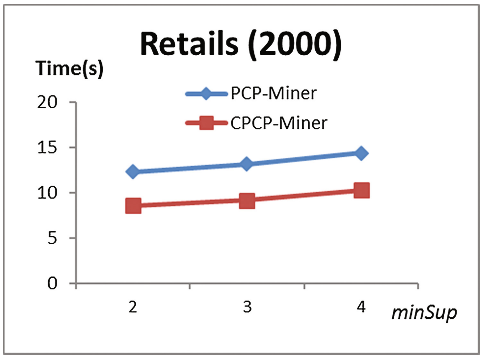


Fig. 2. Comparison of two methods in Retails database with the number of transaction is 2000

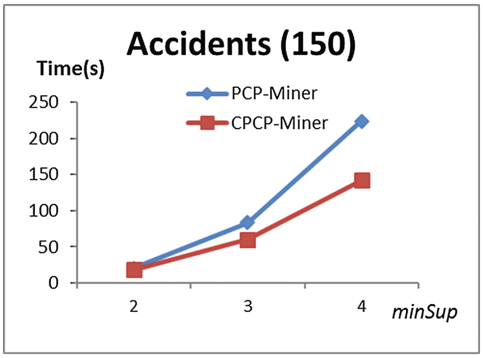


Fig. 3. Comparison of two methods in Accidents database with the number of transaction is 150

Figures 2, 3, 4 and 5 present the results from experimental databases with some *minSup* values (We random items in X with $|X| = 3$ for all experiments).

Results show that CPCP-Miner is always more efficient than PCP-Miner with post-processing. For example, consider Accidents database with *minSup* = 4, the number of transactions is 150 and the number of items is 773, the runtime of CPCP-Miner is 141.9 s while that of PCP-Miner with post-processing is 223.8 s.

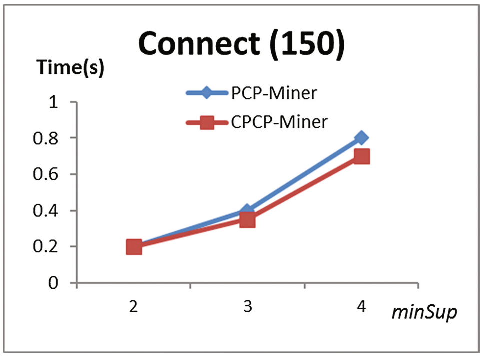


Fig. 4. Comparison of two methods in Connect database with the number of transaction is 150

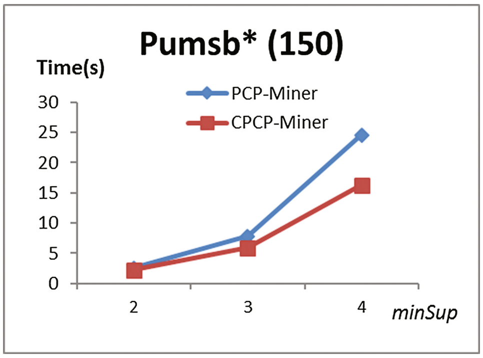


Fig. 5. Comparison of two methods in Pumsb* database with the number of transaction is 150

6 Conclusions and Future Work

This paper has proposed a new method for mining colossal patterns. First of all, the CP-tree structure is developed. Then, Theorem 2 for quickly mining colossal patterns and pruning nodes is designed. Based on these theorems, we propose the CP-Miner algorithm for mining colossal patterns. CP-Miner mines colossal patterns using the CP-tree and pre-processing techniques to reduce the search space. Based on CP-Miner, a theorem for quickly checking duplicated patterns and one for rapidly checking non-colossal patterns are developed. These two theorems provide support for PCP-Miner, an improved of CP-Miner algorithm, which is able to efficiently mine colossal patterns.

One of the weaknesses of bit vectors is that when the database is very sparse and the first position of bit 1 at the beginning and the last position of bit 1 at the end of each bit vector is very far, the number of bits in each bit vector is very large, although it contains a small number of bits 1. Although we have used DBV to solve this problem it only removes bits 0 at the beginning and the end. In future work, we will study how to compress this case of databases in the tree. We will also apply the CP-tree to mining frequent closed patterns and frequent maximal patterns in high dimensional databases.

Acknowledgments. This research is funded by NTTU Foundation for Science and Technology Development under grant number 2017.01.75

This work was carried out during the tenure of an ERCIM 'Alain Bensoussan' Fellowship Programme.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases, In: SIGMOD, pp. 207–216 (1993)
2. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: VLDB 1994, pp. 487–499 (1994)
3. Ao, X., Luo, P., Li, C., Zhuang, F., He, Q.: Online frequent episode mining. In: ICDE 2015, pp. 891–902 (2015)
4. Badia, A., Kantardzic, M.: Generalizing association rules: theoretical framework and implementation. intelligent systems design and applications. In: Advances in Soft Computing, vol. 23, pp. 283–292. Springer, Heidelberg (2003)
5. Baralis, E., Cagliero, L., Cerquitelli, T., Garza, P.: Generalized association rule mining with constraints. Inf. Sci. **194**, 68–84 (2012)
6. Berkhin, P., Dhillon, I.: Knowledge discovery: clustering. In: Encyclopedia of Complexity and Systems Science, pp. 5051–5064 (2009)
7. Cagliero, L., Garza, P.: Improving classification models with taxonomy information. Data Knowl. Eng. **86**, 85–101 (2013)
8. Dabbiru, M., Shashi, M.: An efficient approach to colossal pattern mining. Int. J. Comput. Sci. Network Secur. **6**, 304–312 (2010)
9. Duong, H.V., Truong, T.C., Vo, B.: An efficient method for mining frequent itemsets with double constraints. Eng. Appl. of AI **27**, 148–154 (2014)
10. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: SIGMOD 2000, pp. 1–12 (2000)
11. Hyeok, K., Cholyong, J., Unhyok, R.: Implementation of Association Rule Mining for Network Intrusion Detection. CoRR abs/1601.05335 (2016)
12. Lin, W.Y., Huang, K.W., Wu, C.A.: MCFPTree: An FP-tree-based algorithm for multi-constraint patterns discovery. Int. J. Bus. Intell. Data Mining **5**, 231–246 (2010)
13. Liu, H., Wu, X., Zhang, S.: A new supervised feature selection method for pattern classification. Comput. Intell. **30**(2), 342–361 (2014)
14. Luo, A., Jia, X., Shang, L., Gao, Y., Yang, Y.: Granular-based partial periodic pattern discovery over time series data. In: Rough Sets and Knowledge Technology, RSKT. LNCS, vol. 6954, pp. 706–711. Springer, Heidelberg (2011)
15. Mooney, C., Roddick, J.F.: Sequential pattern mining - approaches and algorithms. ACM Comput. Surv. **45**(2), 1–19 (2013)
16. Ng, R.T., Lakshmanan, L.V.S., Han, J., Pang, A.: Exploratory mining and pruning optimizations of constrained associations rules. In: ACM SIGMOD International Conference on Management of Data, pp. 13–24 (1998)
17. Nguyen, T.L., Vo, B., Snásel, V.: Efficient algorithms for mining colossal patterns in high dimensional databases. Knowl.-Based Syst. **122**, 75–89 (2017)
18. Nguyen, D., Vo, B., Le, B.: CCAR: An efficient method for mining class association rules with itemset constraints. Eng. Appl. of AI **37**, 115–124 (2015)
19. Nguyen, D., Nguyen, L.T.T., Vo, B., Hong, T.P.: A novel method for constrained class association rule mining. Inf. Sci. **320**, 107–125 (2015)

20. Norouzi, M., Bengio, S., Chen, Z., Jaitly, N., Schuster, M., Wu, Y., Schuurmans, D.: Reward augmented maximum likelihood for neural structured prediction. In: NIPS 2016, pp. 1723–1731 (2016)
21. Raorane, A.A., Kulkarni, R.V., Jitkar, B.D.: Association rule – extracting knowledge using market basket analysis. Res. J. Recent Sci. **1**(2), 19–27 (2012)
22. Raza, K.: Application of data mining in bioinformatics. Indian J. Comput. Sci. Eng. **1**(2), 114–118 (2013)
23. Ng, R., Lakshmanan, L.V.S., Han, J., Pang, A.: Exploratory mining and pruning optimizations of constrained associations rules. In: ACM SIGMOD International Conference on Management of Data (1998)
24. Slimani, T., Lazzez, A.: Sequential mining: patterns and algorithms analysis. Int. J. Comput. Electron. Res. **2**(5), 639–647 (2013)
25. Sohrabi, M.K., Barforoush, A.A.: Efficient colossal pattern mining in high dimensional datasets. Knowl. Based Syst. **33**, 41–52 (2012)
26. Srikant, R., Vu, Q., Agrawal, R.: Mining association rules with item constraints. In: Paper presented at the 3rd International Conference on Knowledge Discovery and Data Mining (KDD 1997) (1997)
27. Tran, A.N., Duong, H.V., Truong, T.C., Le, B.H.: Efficient algorithms for mining frequent itemsets with constraint. In: Knowledge and Systems Engineering (KSE), pp. 19–25 (2011)
28. Vo, B., Le, T., Pedrycz, W., Nguyen, G., Baik, S.W.: Mining erasable itemsets with subset and superset itemset constraints. Expert Syst. Appl. **69**, 50–61 (2017)
29. Weichbroth, P., Owoc, M., Pleszkun, M.: Web user navigation patterns discovery from WWW server log files. In: FedCSIS 2012, pp. 1177–1176 (2012)
30. Wojciechowski, M., Zakrzewicz, M.: Dataset filtering techniques in constraint-based frequent pattern mining. Pattern Detect. Discov. **2447**, 77–91 (2002)
31. Zaki, M.J., Parthasarathy, S., Ogihara, M., Li, W.: New algorithms for fast discovery of association rules. In: KDD 1997, pp. 283–286 (1997)
32. Zaki, M.J., Hsiao, C.J.: Efficient algorithms for mining closed itemsets and their lattice structure. IEEE Trans. Knowl. Data Eng. **17**(4), 462–478 (2005)
33. Zhang, C., Zhang, S.: Association Rule Mining: Models and Algorithms. LNCS, vol. 2307. Springer (2002). ISBN: 3-540-43533-6
34. Zhang, S., Wu, X.: Fundamentals of association rules in data mining and knowledge discovery. Wiley Interdisc. Rev. Data Mining Knowl. Discov. **1**(2), 97–116 (2011)
35. Zhou, W., Liu, H., Cheng, H.: Mining closed episodes from event sequences efficiently. PAKDD **1**, 310–318 (2010)
36. Zhu, F., Yan, X., Han, J., Yu, P., Cheng, H.: Mining colossal frequent patterns by core pattern fusion. In: ICDE 2007, pp. 706–715 (2007)
37. Zubi, Z.S., Raiani, M.S.E.: Using web logs dataset via web mining for user behavior understanding. Int. J. Comput. Commun. **8**, 103–111 (2014)

A Dynamic Packed Approach for Analytic Data Warehouse in Ad-Hoc Queries

Loan T.T. Nguyen[1], Hung Son Nguyen[2](✉), and Sinh Hoa Nguyen[3]

[1] VŠB-Technical University of Ostrava, Poruba, Ostrava, Czech Republic
nthithuyloan@gmail.com

[2] Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, Warsaw, Poland
son@mimuw.edu.pl

[3] Polish-Japanese Academy of Information Technology, Koszykowa 86, 02-008 Warsaw, Poland
hoa@mimuw.edu.pl

Abstract. Brighthouse is a column-oriented data warehouse that supports the compressed databases as well as analytic querying. For the faster query processing, Brighthouse creates packages from the data rows. While the query is resolving, it decompresses only those packages that partially satisfy the condition of the query to avoid accessing all the database. However, Brighthouse used a constant parameter to create packages, this may create incompact packages and lead to large number of packages that are processed in each query. In this paper, at first, we define the task of partitioning data table into blocks as an optimization problem, then discuss the time complexity of the problem and propose an efficient algorithm, which creates dynamically data packages for efficient queries in databases. The experimental results shown the advantage of the proposed approach in package range reduction.

Keywords: Bright-house · Data pack · Sequence of time · Compress data

1 Introduction

In traditional data warehouse, tables are stored in row-oriented format, known as the N-ary storage model. Tables can also be stored in column-oriented model. The last format optimizes performance for SQL queries that involve a small set of columns. However, data updates or join operations in multiple columns architecture become time-consuming. A compromise between these two models is to divide a data table horizontally into blocks. Within blocks records are stored in column-oriented format. The idea was introduced in the PAX (Partition Attributes Across) architecture. In PAX model [1] a table is horizontally partitioned into pages. In each page, all values of each attribute are grouped together, which greatly improves cache performance. The idea of automatic creation and usage of higher-level data about data has also developed in the Brighthouse [8]. In the InfoBright project, the authors have proposed an algorithm to generate metadata, which helps optimize the queries by reducing data row access when calculate the query values. Knowledge Nodes are used in place of the traditional query indexing methods and thus reduce the storage space and runtime needed to calculate query values. In this approach, the authors referred database compression in stated-of-the-art [6, 7, 9, 10]. This method has an interesting point, it uses rough set to quickly determine the relevant/suspect/irrelevant data packs.

Other similar approaches have been discussed in [2–5]. The common feature of the proposed approaches is that it breaks a data table into blocks with a fixed size, independently of the data distribution. It can show drawback when data have a random distribution. This leads to more suspect packs presented inside the queries and thus slow down the querying time, since the queries must load physical data from these packs. In this paper, we propose an efficient method to create the data packs based on the data distribution. The partition task can be defined as the clustering problem with some additional constraints. There are different heuristics solving the problem. In the paper. we propose a linear heuristic, which can split stream data into blocks in dynamical manner.

First, we normalize the difference between the *min* and *max* value for adjacent data values by column. Next, on the i^{th} row, we search for the max value from the normalized values and call it n_i. If *max* value of a row is less than or equal to a specified threshold ξ, that row and its following row are belonged to the same pack, i.e., we only split pack when $n_i > \xi$. With this method, we adjust the threshold value ξ to balance between the metadata storage space and limit the physical data stream reading (assume the data has a very large number of data rows).

The rest of the paper is organized as follows: Sect. 2 presents problem related to packed approach. Section 3 describes the computational complexity of the packing problem. Section 4 presents the proposed algorithm and an example is also given while Sect. 5 shows experimental results on the standard database. Finally, conclusions and future works are described in Sect. 6.

2 Partition Problem Formulation

The problem of splitting data table into *continuous blocks* that minimize range within blocks and maximize a gap among blocks can be defined as clustering problem with some constraints.

Definition 1 (Data table)
Data table is a triple $D = (U, Id, A)$, where U is a finite set of records and A is a set of attributes and Id is the indexing function or briefly the index. Formally, if $|A| = n$ and $|U| = m$, the index is the bijection: Id: $U \rightarrow \{1, 2,\ldots, m\}$ and each attribute a_i is a function a_i: $U \rightarrow R$ for all $a_i \in A$.

Definition 2 (Partition)
Let $D = (U, Id, A)$ be a given data table. The partition of U into k blocks is a family of subsets of U, which satisfied the following conditions:

- $U = U_1 \cup U_2 \cup \ldots \cup U_k$;
- $\forall_{x,y}\ (x \in U_i) \wedge (y \in U_j) \wedge (i < j) \rightarrow Id(x) < Id(y)$

Definition 3 (Range of block and total range of partition)
Let $D = (U, Id, A)$ be a given data table. For any $a_i \in A$ and subset $U_j \subseteq U$, the normalized range of attribute a_i within subset U_j is defined as follows:

$$Range\left(a_i, U_j\right) = \frac{\max\left(a_i\left(U_j\right)\right) - \min\left(a_i\left(U_j\right)\right)}{\max(a_i(U)) - \min(a_i(U))}$$

Let $U_j \subset U$. The range of block U_j is defined as:

$$Range\left(U_j\right) = \sum_i Range\left(a_i, U_j\right)$$

The total range of partition $U = U_1 \cup U_2 \cup \ldots \cup U_k$ is defined as:

$$TotalRange(U) = \sum_{i=1}^{k} Range(U_i)$$

The optimal partition problem can be defined as follows:

Partition problem:
Given: Data table $D = (U, Id, A)$ and a constant k.
Task: Divide U into k blocks $U_1 \cup U_2 \cup \ldots \cup U_k$ with *minimal* total range.

Let us notice the similarity between the partition problem and the clustering problem. The difference is that in the packing problem the order of records is remains.

3 Computational Complexity of the Partition Problem

The exact (optimal) algorithm for the partition problem has the following schema:

Step 1: Create all possible of partitions of *Id* set into k disjoint blocks.
Step 2: For any partition P calculate $TotalRange(P)$.
Step 3: Choose the partition with the *min TotalRange*.

Theorem: The exact algorithm runs in time $O(n^k)$.

Proof: The problem of partition of an *n-element* set into k disjoint blocks is equivalent to the problem of *distributing n identical objects into k distinct boxes*. It is known that the number of ways to distribute the objects is:

$$C_{n+k-1}^{k-1} = \frac{(n+k-1)!}{(k-1)!n!} = \frac{(n+1)(n+2)\ldots(n+k-1)}{k!} = O\left(n^k\right)$$

Unfortunately, when database contains millions of records, the exact algorithm is useless. In the next section, we propose heuristics, which in linear time generate a sub-optimal solution.

4 Proposed Method

In this section, we present a method to create packs based on a threshold ξ. To do that, following equation is used to normalize the gap between min and max values of each row.

$$n_i = \max\left\{\frac{\max\left(v_{ij}, v_{i+1j}\right) - \min\left(v_{ij}, v_{i+1j}\right)}{\sum_{v_{ij}\in Col_j} v_{ij}}\right\}_{j=1}^{n} \tag{1}$$

where v_{ij} is value at row i, column j of the matrix and Col_j is the column j^{th}.

4.1 Algorithm

The purpose of this problem is to find a best splitting to partition data rows into packs such that we can reduce reading physical data. Assume that we want to divide data rows into k packs. To solve this problem, we present a heuristic method, named Dynamic Pack Approach (DPA), to partition data into packs based on a threshold ξ.

Input: Database (D) with m rows and n columns, $\xi > 0$
Output: packs to compress data rows for query
DPA (D, ξ): Dynamic Pack Approach Algorithm
Step 1: Compute *min* and *max* value of two adjacent rows
Step 2: Compute the normalize value (n_i with $i \in [1, m\text{-}1]$) of each row using Eq. (1)
Step 3: Splitting data rows into each pack based on n and ξ
Step 4: Computing values for each pack.

4.2 An Illustration of Example

In this section, we use the database in Table 1 to illustrates the process of DPA with ξ = 0.05.

Table 1. An example training dataset

ID	A	B
1	100	2000
2	50	1800
3	200	2100
4	50	1900
5	1500	20
6	1000	40
7	500	60
8	1000	50
9	50	1000
10	100	800

(*continued*)

Table 1. (*continued*)

ID	A	B
11	80	1100
12	60	1000
13	120	900
14	400	90
15	300	120
16	480	100
17	500	110
18	1500	600
19	1400	500
20	1300	550

Table 1 has 20 rows containing two attributes *A* and *B*. Assume that we compute the min and max between two adjacent rows such as {1,2}, {2,3}, …, {19,20}. We have the results in Table 2.

Table 2. Min and max of attributes *A* and *B*

Two adjacent rows	Min A	Max A	Min B	Max B
{1,2}	50	100	1800	2000
{2,3}	50	200	1800	2100
{3,4}	50	200	1900	2100
{4,5}	50	1500	20	1900
{5,6}	1000	1500	20	40
{6,7}	500	1000	40	60
{7,8}	500	1000	50	60
{8,9}	50	1000	50	1000
{9,10}	50	100	800	1000
{10,11}	80	100	800	1100
{11,12}	60	80	1000	1100
{12,13}	60	120	900	1000
{13,14}	120	400	90	900
{14,15}	300	400	90	120
{15,16}	300	480	100	120
{16,17}	480	500	100	110
{17,18}	500	1500	110	600
{18,19}	1400	1500	500	600
{19,20}	1300	1400	500	550

For example, min(a_1, a_2) = 50 and min(a_5, a_6) = 1000; similar max(a_1, a_2) = 100 and max(a_4, a_5) = 1500, respectively. Then, we compute n_i for each row and the results are shown in Table 3.

Table 3. The gaps bigger then threshold ξ

Two adjacent rows	n_i
{1,2}	0.013477089
{2,3}	0.020215633
{3,4}	0.014031805
{4,5}	**0.135640786**
{5,6}	0.046772685
{6,7}	0.046772685
{7,8}	0.046772685
{8,9}	**0.088868101**
{9,10}	0.013477089
{10,11}	0.020215633
{11,12}	0.006738544
{12,13}	0.006738544
{13,14}	**0.05458221**
{14,15}	0.009354537
{15,16}	0.016838167
{16,17}	0.001870907
{17,18}	**0.09354537**
{18,19}	0.009354537
{19,20}	0.009354537

For example, consider $\xi = 0.05$, the values of the first three rows are smaller than ξ, so rows 1, 2, 3, and 4 are in same pack (Pack 1). At row fourth, we have $n_4 > \xi$, it means rows 4 and 5 do not belong to the same pack (i.e., row 5 belongs to Pack 2). Do the same way, because n_8, n_{13} *and* n_{17} are greater than ξ, we have 5 packs as follows: Pack 1 contains rows 1, 2, 3, and 4; Pack 2 contains rows 5, 6, 7, and 8; Pack 3 contains rows 9, 10, 11, 12, and 13; Pack 4 contains rows 14, 15, 16, and 17; and Pack 5 contains rows 18, 19, and 20.

5 Experiments

The experiments were implemented by C# in .NET 2010 environment on a PC with following configuration: Windows 8.1, CPU core i7-7500U 2.70 GHz, RAM 8 GB.

Experiments are evaluated on the real database (*Coal mine*) and their characteristics are showed in Table 4.

Table 4. Characteristics of experimental databases

Database	#attributes	#rows
DatabaseTest	2	20
CoalMine-1	2	3,860
CoalMine-2	2	79,893

Table 5 shows the numbers of packs and rows in each pack in experimental databases with some thresholds.

Table 5. Number of packs and rows in each pack

Database	ξ	Number of packs	Rows belong to each pack
DatabaseTest	0.05	5	1..4; 5..8; 9..13; 14..17; 18..20
CoalMine_1	0.05	5	1..2555; 2556; 2557..3844; 3845; 3846..3760
CoalMine_2	0.02	5	1..416; 417; 418..37738; 37739..37761; 37762..79893

To show the efficient of proposed algorithm, we compute the sum of the distances between max and min values in each column in each pack. Figures 1 2, 3, 4, 5 and 6 show the comparison of DPA and the method based on partition the data table into equal-frequency packs presented in Slezak et al. [8].

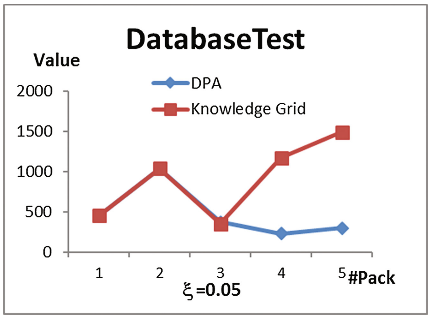


Fig. 1. Comparison of two methods in DatabaseTest with ξ = 0.05

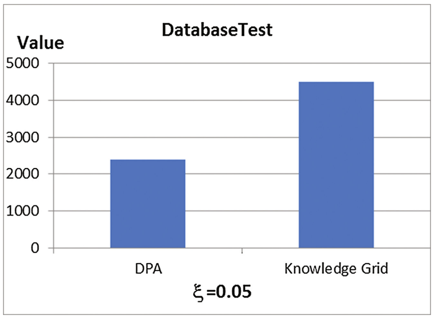


Fig. 2. Sums of all values from packs of two methods in DatabaseTest with ξ = 0.05

Results from Fig. 1 shows that DPA is better than other method. For the first two packs, they have the same rows, so the sums of (max-min) are the same. For pack 3, the sum of DPA is 370 while that of Slezak et al. method is 350. It means our method is not better than Slezak et al. method. However, for the rest packs, the corresponding values of DPA are 230, 300 while that of Slezak et al. are 1170, 1490. Figure 2 shows the sums of all values from packs of two methods.

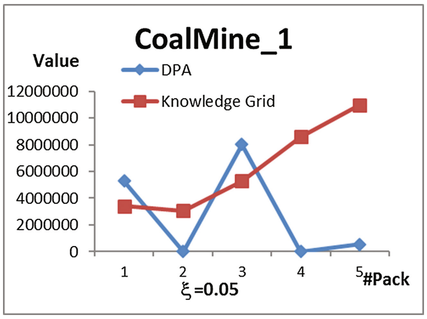


Fig. 3. Comparison of two methods in CoalMine-1 with ξ = 0.05

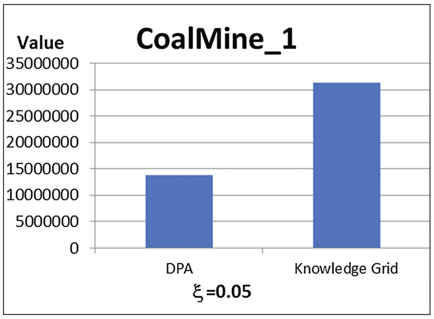


Fig. 4. Sums of all values from packs of two methods in CoalMine-1 with ξ = 0.05

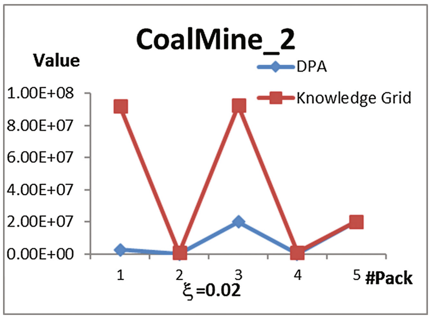


Fig. 5. Comparison of two methods in CoalMine-2 with ξ = 0.02

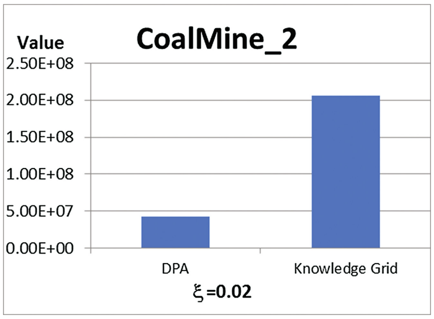


Fig. 6. Sums of all values from packs of two methods in CoalMine-2 with ξ = 0.02

6 Conclusions and Future Work

In this paper, we have proposed an efficient method to partition data with a dynamic creating data packs based on the normalization values between two adjacent rows. This method can solve the problem of chosen the number of packs and the gap of values in each pack is not large. This leads to reduce scanning physical data rows in each query. In future work, we are going to compute the runtime for queries and compare our method with Bright-house. In addition, we plan to investigate how to use the compress database for mining instead using the original database.

Acknowledgments. This research is partially supported by the project “Parking space in rest and service areas (RSA)” financed by NCBiR/GDKKiA as a part of common undertaking “RID”, under the contract DZP/RID-I-44/8/NCBR/2016.

This work was carried out during the tenure of an ERCIM ‘Alain Bensoussan’ Fellowship Programme.

References

1. Ailamaki, A., DeWitt, D.J., Hill, M.D.: Data page layouts for relational databases on deep memory hierarchies. VLDB J. **11**(3), 198–215 (2002)
2. Apaydin, T., Canahuate, G., Ferhatosmanoglu, H., Tosun, A.S.: Approximate encoding for direct access and query processing over compressed bitmaps. In: VLDB, pp. 846–857 (2006)
3. Beyer, K.S., Haas, P.J., Reinwald, B., Sismanis, Y., Gemulla, R.: On synopses for distinct-value estimation under multiset operations. In: SIGMOD, pp. 199–210 (2007)
4. Bruno, N., Chaudhuri, S., Gravano, L.: STHoles: a multidimensional workload aware histogram. In: SIGMOD, pp. 211–222 (2001)
5. Chakkappen, S., Cruanes, T., Dageville, B., Jiang, L., Shaft, U., Su, H., Zait, M.: Efficient and scalable statistics gathering for large databases in Oracle 11g. In: SIGMOD, pp. 1053–1063 (2008)
6. Ferragina, P., Grossi, R., Gupta, A., Shah, R., Vitter, J.S.: On searching compressed string collections cache-obliviously. In: PODS, pp. 181–190 (2008)
7. Holloway, A.L., Raman, V., Swart, G., DeWitt, D.J.: How to barter bits for chronons: compression and bandwidth tradeoffs for database scans. In: SIGMOD, pp. 389–400 (2007)
8. Slezak, D., Wroblewski, J., Eastwood, V., Synak, P.: Brighthouse: an analytic data warehouse for ad-hoc queries. PVLDB **1**(2), 1337–1345 (2008)
9. Vo, B., Manku, G.S.: RadixZip: linear-time compression of token streams. VLDB **2007**, 1162–1172 (2007)
10. Zukowski, M., Heman, S., Nes, N., Boncz, P.A.: Super-scalar RAM-CPU cache compression. In: ICDE, p. 59 (2006)

Internet of Things, Cloud Computing and High Performance Computing, Distributed Computer Systems, Content Delivery Networks

How Is Server Software Configured? Examining the Structure of Configuration Files

Błażej Święcicki(✉) and Leszek Borzemski

Faculty of Computer Science and Management,
Wroclaw University of Science and Technology, Wrocław, Poland
{blazej.swiecicki,leszek.borzemski}@pwr.edu.pl

Abstract. Contemporary software on servers is usually configured through editing configuration files. Surprisingly, to the extent of our knowledge, the diversity of configuration file formats remains unstudied. Our goal in this work is to determine what data structures are used in configuration files and what formats are being used to encode them, in order to have a foundation for semantic analysis of their contents in the future. We examine 14133 files in 3409 packages comprising the whole set of software available in Debian stable repositories that has configuration files in the /etc directory. After eliminating files that are not configuration (such as init scripts), we assign them to categories using various criteria, some of them being the data structure they express or whether the order of statements matter. In this examination we find that even custom configuration formats can usually be expressed using one of several commonly used data structures. Some software packages, however, are configured in a Turing-complete programming language, usually the same one that the configured program was written in, and are therefore unsuitable for static analysis. Regardless, we provide a taxonomy of configuration formats, and highlight common themes in custom formats used by various packages. Ultimately, we describe a data structure that is able to hold information about all of these configuration files for further analysis.

Keywords: Configuration management automation · Data modeling · Configuration · Static analysis · System configuration

1 Introduction

Contemporary software on servers is usually configured through editing configuration files. These files are instructions on how a configured program should behave, and usually, but not always, can be thought of as data structures. Being data structures, they are static, or at least not Turing-complete, and it is possible to analyze and modify their contents without running the configured program or any other program dedicated for this purpose.

There is another meaning to the word *configuration*, usually referred to as runtime configuration - that is, the contents of the aforementioned data structures during the execution of a program. Sometimes, this runtime configuration can be changed without modifying the corresponding configuration files. We ignore that fact completely and

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 38th International Conference on Information Systems Architecture and Technology – ISAT 2017*,
Advances in Intelligent Systems and Computing 655, DOI 10.1007/978-3-319-67220-5_20

focus exclusively on configuration files. The reason is that it is not always possible to extract runtime configuration from a running program, when it is, every program has a different method of accessing it, and regardless, it is considered a good practice to reflect any runtime changes in the configuration files, so that it may survive restarting the service, or rebooting the whole server.

Our ultimate goal is to provide a method for automatic verification of correctness of a whole system's configuration, in contrast to just checking one configuration file. This is motivated by the fact that a lot of software systems' failures are caused by misconfigurations [2, 3, 9, 10, 18]. Although there are many different approaches to alleviate that problem, some based on automatically generating and managing configuration files [11–17], others on verification of existing configurations [5, 8, 9], instrumenting configured applications to detect errors earlier [4], and static analysis of the program's source code to detect potential errors [19], the problem is not yet solved [6, 7]. In automated configuration management, it would be useful to have means to identify potential problems introduced by it. This publication is a first step towards a new approach to the problem, detailing what kinds of data can be expected in configuration files. Therefore, this analysis is in context of later verification of configuration – we do not consider, for example, how to transform data back into the text of a configuration file.

Our contribution comprises three parts: we define a taxonomy of configuration files, list classes of errors that might arise in these files, and propose a data structure that can hold the configuration described in these files in a way that allows for further analysis of whether any of these errors have occurred.

In this article, we first present the methodology of our research in Sect. 2, then discuss findings in Sect. 3. Later, we move on to describing our proposed taxonomy in Sect. 4, discuss the possible error classes in Sect. 5. In Sect. 6, we define a data structure that can contain configuration data found in the analyzed configuration files while allowing to check for error classes described in Sect. 4. At last, in Sect. 7 we draw conclusions.

2 Methodology

We begin by searching for all packages in Debian jessie (the current stable release) that contain configuration files. This is accomplished in the following way:

```
apt-file search -x '/etc/.*' | grep ': /etc' | grep -vP
': /etc/init.d' | cut -d: -f1 | uniq
```

We filter out packages containing only files in `/etc/init.d`, as these files are not configuration, and including these packages would be useless. We search for files in `/etc`, because it is the standard place for configuration in Unix systems. This yields 14133 files in 3409 packages.

Each of these packages is then downloaded, and extracted. We then analyze each file manually, and examine differences between them. We have chosen to answer the following questions for all configuration files:

1. Does order of configuration entries matter?
2. What type of structure does the file use?
3. Do the values in the file have complex types?
4. Do values reference other parts of the file?
5. What syntax or programming language is the configuration written in?
6. What is the syntax for comments, if they are possible?

3 Findings

One of the most easily visible aspects of a configuration file is its syntax. We have found that most configuration files employ one of several popular configuration formats to express their configuration. The two most popular formats were XML and INI. INI files do not have a formally specified syntax, and different programs vary in some specifics. Consider the short examples shown in Fig. 1.

```
# Comment
a key = value
[section]
b key = value
```

```
; Comment
[general]
a_key = value
[section]
b_key = "value"
```

```
# Comment
a_key value
b_key value
```

Fig. 1. Three examples of configuration files that contain key-value data

The differences are mostly in whether there is an implicit "global" section, what character is used for comments (which can be easily inferred by checking whether the file contains lines starting with either # or;), and whether the = character is used to separate the key from its value. Sometimes, quotes are used to denote the beginning and end of a value, and they should not be treated as a part of it. Note that the last example wouldn't be traditionally considered an INI file, but it has the same structure and semantics as other configuration files. These files describe configuration as a list key-value pairs, where the keys are strings, and values vary in their type [1].

One important difference between different programs and sometimes even statements in a configuration file is whether multiple values for a key are allowed, and how are they expressed. Consider the configuration file for the Linux DNS resolver (`/etc/resolv.conf`), as shown in Fig. 2.

```
search example.org example.com
nameserver 8.8.8.8
nameserver 8.8.4.4
```

Fig. 2. Linux dns resolver configuration

This file uses two methods to denote multiple values: in the case of search, the value contains multiple space-separated entries, while in the case of nameserver there are two keys with the same name, but different values. In both cases, the order of the values matters. In other programs' configuration files, multiple keys may not be allowed, and if they are, the order might not matter. Unfortunately, this is not something one can infer just from analyzing the contents of a file.

Figure 3 shows that the same issue may arise even in XML files.

```
<rule id="r1" order="allow,deny">
    <allow id="a1">
        dacs_admin()
    </allow>
</rule>
```

Fig. 3. A fragment of an xml configuration file containing a non-scalar value in an attribute

In this example (taken from the dacs package), they can be many <rule> tags, and their order matters, but there is also the order attribute. This attribute contains a list of values, separated by a comma. Therefore, an assumption that, even in XML files, values are scalar would be incorrect.

Which leads to another differentiating factor between configuration files: what kinds of values are allowed?

Many configuration files accept only numbers and string values that have no intrinsic meaning, such as names for various things. These are scalar values - they usually cannot be decomposed further. However, some values are filesystem paths, URIs, IP addresses, or even more complex data structures.

For example, Fig. 4 shows a line from a Systemd mount unit that contains a key called Options, with a value that is a list of key-value pairs. Their syntax is almost the same as an INI file, except that commas are used instead of newlines, and some keys have no values, as they are flags. The values in this list can be further decomposed. For example, the value of lowerdir is a list of filesystem paths. Even umask, despite it appearing to be a number, can be decomposed to reveal its true meaning: that there are no restrictions on the owner and group of files in this mount, but others can't execute them. This is because filesystem permissions in Unix systems are concisely expressed as a number in which every bit has its separate meaning.

Therefore, as we can see, any program attempting to analyze the semantics of configuration files must be able to understand not only the syntax of the file itself, but also what is the type of values for given configuration parameters - if only to decompose them into smaller components. Sometimes, as seen above, this knowledge must extend beyond what the configured program knows about the value. In the example of the mount options, Systemd does not understand the contents of these; it just passes them to the mount utility, which in turn passes them to the filesystem driver without further analysis. However, what is usually important for a system administrator configuring such a thing is whether it will ultimately do what's expected, instead of, for example, returning an error.

```
Options =
  lowerdir=
    /mnt/a:
    /mnt/b,
  upperdir=/mnt/c,
  workdir=/mnt/d,
  user=nobody,
  umask=001,
  ro
```

Fig. 4. A fragment of a system mount unit configuration. Note that this is a single line, and is split into multiple in order to show the hierarchy of parameters

There is one more special case that can be encountered with string values: they might contain references to variables defined elsewhere in the configuration file or available by default. One program that allows for this is Nginx.

Figure 5 shows a fragment of a configuration file for Nginx. Aside from the fact that it is not a simple key-value configuration file (which will be discussed later), it shows the use of variables in strings. There are two challenges that must be overcome when analyzing these values. Firstly, while the strings contain references to variables, they are not composed of them - they might contain other text as well. Because of that, both an analysis of the string as a whole must be possible, and checking what variables are used (and whether their type is correct, if applicable). Secondly, the `domain` variable is defined by a capture group in a regular expression. In order to know that, one must be able to parse the particular regular expressions flavor used by Nginx. Similar challenges arise in configuring other programs. It has to be noted, however, that it is still possible to determine what variables are available - this might not be a case with programs configured in various programming languages, such as Python, Ruby, or Javascript.

Some configuration files need a more complex structure than a key-value store would allow. Web servers often have such configuration files, one example being haproxy (Fig. 6).

```
server {
  server_name "~^(www\.)?(?<domain>.*)$";
  location / {
    fastcgi_split_path_info ^(.+\.php)(/.*)$;
    fastcgi_param SCRIPT_FILENAME $document_root
$fastcgi_script_name;
    fastcgi_param HTTP_X_DOMAIN $domain;
  }
}
```

Fig. 5. A fragment of a configuration file for Nginx. Line 5 (`SCRIPT_FILENAME`) is wrapped due to space constraints.

```
frontend front
bind 0.0.0.0:80
bind 0.0.0.0:443 ssl crt /etc/haproxy/ssl.pem
option forwardfor except 172.16.0.0/12
reqadd X-Forwarded-Proto:\ https if { ssl_fc }
acl is_authenticated http_auth(users)
http-request auth realm restricted\ area
  if !is_authenticated { ssl_fc }
```

Fig. 6. A fragment of a haproxy configuration. The last line is wrapped due to space constraints.

This seemingly simple example is quite difficult to analyze semantically, compared to many other configuration files. Firstly, the first line opens a section. However, there's no statement closing a section, as the only method to do so is to open another section. This is similar to INI files, but unlike those, there are several keywords that start a new section. Because of that, even a partial parser would have to know all of them. Another aspect is that many of the options take optional arguments. For example, `bind` takes `ssl` and `crt <path>` as options, and the latter is semantically valid only if the former is used, that is - it is an error to use `crt` without `ssl`. Then, there is the `option` keyword, which allows one to set various flags in haproxy. The issue here is that what is being set is not exactly a flag - it has additional data in the form of `except <address>` that is only valid in the `forwardfor` option. Haproxy, in its documentation, decides to treat `option forwardford` and other options as separate statements, that is, `option` itself does not have any meaning. There's also a question of how much to parse in the `reqadd` statement. Its value should be a valid http header, but haproxy will accept any string. Additionally, since haproxy allows for using variables inside strings, it might be undecidable whether it is a correct http header. Finally, `if` options are supported in several statements. These options contain one of more predicates, of which all must be true for the statement to take effect. Each predicate is either a rule name, which must be defined earlier, and not later, in the file, or an ad-hoc specification for a rule. Because the order of defining rules matters, an analysis engine must know what the current contents of the rule are at the time of its use. When analyzing such complex files, an analysis engine must not ignore these details and be able to express the complex data structures seen here, for example "maybe a http header" in `reqadd`.

4 Taxonomy of Configuration Files

We have analyzed how configuration files structured in applications available in Debian repositories. We propose a taxonomy to classify them in Fig. 7.

First of all, we discard configuration files based on Turing-complete programming languages. One could parse and try to analyze them if they contain only assignments, but such a subset can be classified in other categories. Files in this category are usually written in the same programming language as the program itself.

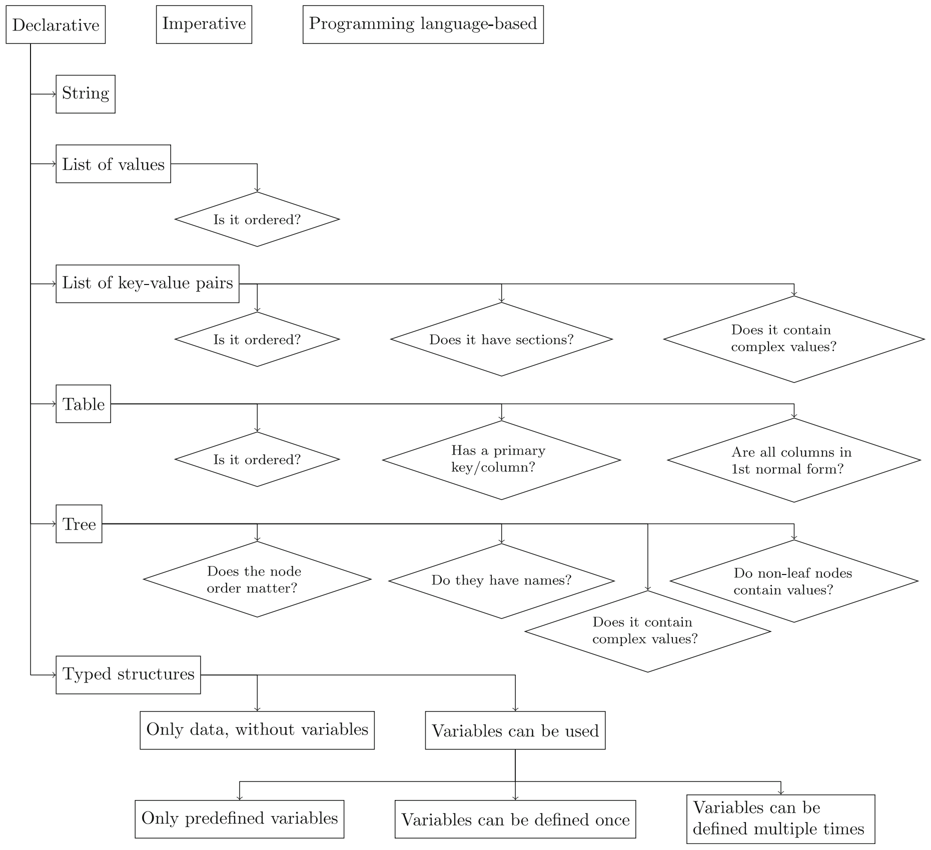

Declarative
Imperative
Programming language-based
String
List of values
Is it ordered?
List of key-value pairs
Is it ordered?
Does it have sections?
Does it contain complex values?
Table
Is it ordered?
Has a primary key/column?
Are all columns in 1st normal form?
Tree
Does the node order matter?
Do they have names?
Does it contain complex values?
Do non-leaf nodes contain values?
Typed structures
Only data, without variables
Variables can be used
Only predefined variables
Variables can be defined once
Variables can be defined multiple times

Fig. 7. The taxonomy of configuration file formats

Next, we make another distinction between imperative and declarative configuration files. An imperative configuration file is a list of instructions (as opposed to a list of data values). An example would be firewall configuration, which is stored as a list of firewall rules to be applied. Each line in such a configuration file is an instruction to add, remove or modify rules in the firewall, modifying its state. Of course, in most configuration files, only additions would be used, but nevertheless, structurally it is a list of instructions, and not just data.

Declarative configuration is easier to analyze. The simplest form of configuration is a single string value. Although rarely, it can be seen in some programs, that usually just display the contents of the file to a user in a certain situation. Examples would be "message of the day" files, or http error pages. Adding the first dimension to the configuration, there are files that contain lists of values. These values are scalars, usually either filesystem paths or regular expressions. One important distinction is whether their order is important or not.

A lot of configuration files store a list of pairs, as it a format both easy to parse and use in a program, and to reason about. Although the syntax varies greatly, the most important semantic difference is whether keys can contain sections and non-scalar values. In this category, keys have predefined names.

If there is a need for more than one value for a key, often a table is used. There are files that contain two column tables. These differ from key-value lists described above in that while there might be a primary key value, it is still a value, and not a name that is defined in the program. In some cases, fields in the table can contain non-scalar values, which is also important for analysis.

A relatively uncommon data structure for a configuration file is a generic tree, as is used in xml. The main differentiator for this category is that there is no method of validating the structure of the tree syntactically. Some programs will report errors when there is unknown data in the tree, others will not. Again, in this case, the values might be non-scalar, as is described in the previous section.

Lastly, there are configuration files that, while declarative, have more complex structures. All of them can be expressed as trees, however their structures are more constrained. In these cases, a given tree node has a type, which decides what types of children it might have. It is important to note that these nodes might also have values in addition to their types. In files in this category, it is often possible to use variables referring to other parts of configuration - either predefined, or defined in the configuration file itself.

5 Errors in Configuration

With this taxonomy, we proceed to define the types of errors that might arise in configuration files. We find the following classes of errors to be possible:

- value equals *X*,
- value does not equal *X*,
- value is less than *X*,
- value is more than *X*,
- wrong order of values,
- invalid syntax,
- invalid key,
- undefined variable (in case where it is possible to define them),
- invalid variable (in case where it is not possible to define new variables),
- presence of a key when it is forbidden by another part of configuration,
- absence of a required key,
- multiple values where only one is allowed,
- more than *X* values for one key,
- less than *X* values for one key,

where *X* can be:

- a simple value,
- a value of another configuration setting, possibly of a different program,
- or a value representing a part of the current state of the configured system (for example its current IP address).

The “key” in these categories is either a key in a list of key-value pairs, a field in a table, or a type of a single node in a tree.

These categories encompass a variety of smaller use cases, such as two services using the same resource on the same system, or having multiple values for a key where only one is allowed.

6 Universal Configuration Description

In order to analyze these configuration files holistically, we must find a common data structure to encode all variations of these files. We focus on declarative configuration and propose the following structure to hold information about the values in configuration files:

```
ConfigValue: (type: String,
              value: Scalar?,
              children: (List<ConfigValue> | Dictionary)?
)
Dictionary: List<(String,ConfigValue)>
Scalar:(value:Number|String,
       usesVariables:List<String>,
       definesVariables:List<String>,
       setsVariables:List<String>
)
```

For further reference, we call this the Universal Configuration Description (UCD). Because complex values exist in various configuration files, the data structure is necessarily recursive. Both `value` and `children` fields of `ConfigValue` are optional, as denoted by “?” at the end of their type declarations. Each value has a type – expressed in terms of the configured program, for example a server configuration in haproxy would have a “HaproxySever” type, and a `Dictionary <String, ConfigValue>` value, because it has multiple named fields that are unordered. Scalar values, in addition to having their content, contain information about the variables they use, define, and set (or redefine), so that a simple analysis of whether they can be used in a given context may be performed. Analysis of the content of values that use variables remains an open question, however, and is out of scope for this document.

With such a structure, a configuration file becomes a single `ConfigValue` containing its content. If the file contains a string or a list, the transformation can be performed directly, as there are appropriate data types in the structure. In the case of key-value pairs, if they contain sections, the file would contain a dictionary with section names as keys and dictionaries as values. The order of keys is always preserved for dictionaries, as it is easily possible to simple ignore it during analysis if it is not important. One important detail is the case of multiple values for one key. Such values shall be always represented as multiple key-value pairs, as it is the only method that can preserve their order relative to other keys. Tables in configuration files are expressed as lists of dictionaries if they don't contain primary keys, and dictionaries of dictionaries if they do. Trees are expressed as nested lists or nested dictionaries, depending on whether the branches have names.

Strings, lists, key-value sets, and trees have the type field set to their respective type names. Tables and any other structures must have their own type names per structure type, so that an analysis engine can check whether its contents match a predefined structure. For example, the `/etc/passwd` containing a table with information about a system's users would have a "Passwd" type, not "Table", as it's a specific table with specific fields. The contents of any complex structure are encoded in the same manner as trees.

In order to explain how the transformation to UCD could be done, we present the transformed version of the configuration fragment found in Fig. 5:

```
ConfigValue {
    type = "NginxServer",
    value = null,
    children = [
        ("server_name", ConfigValue {
            type = "Regexp",
            value = Scalar {
                value="^(www\.)?(?<domain>.*)$",
                usesVariables=[],
                definesVariables=["domain"],
                setsVariables=[]
            },
            children = null
        }), ("location", ConfigValue {
            type = "NginxLocation",
            value = Scalar {
                type="String",
                value="/",
                usesVariables=[],
                definesVariables=[],
                setsVariables=[]
            }, children = [
                ("fastcgi_split_path_info", ConfigValue {
                    type = "Regexp"
                    value=Scalar {
                        value="^(.+\.php)(/.*)$",
                        usesVariables=[],
                              definesVariables=
["fastcgi_script_name", "fastcgi_path_info"],
                              setsVariables=[]
                          }
                          children=null
                      }),
                      ("fastcgi_params", ConfigValue {
                          type = "Dictionary"
                          value=null
                          children = [
                              ("SCRIPT_FILENAME", ConfigValue {
                                  type="String"
                                  value=Scalar {
                                      value=
"$document_root $fastcgi_script_name",
                                      usesVariables=
["document_root", "fastcgi_script_name"],
                                      definesVariables=[],
                                      setsVariables=[]
                                  }
                              }),
                              ("HTTP_X_DOMAIN", ConfigValue {
                                  type="String"
                                  value=Scalar {
                                      value="$domain",
                                      usesVariables=["domain"],
                                      definesVariables=[],
                                      setsVariables=[]
                                  }
                              }),
                          ]
                      }),
                  ]
              })
          ]
}
```

It should be noted that both the `server_name` and `fastcgi_path_info` settings define variables, although the reasons for that are different: `server_name` specifies a name for a regular expression group that can be used later as a variable, while `fastcgi_path_info` always defines its variables, even though they are not mentioned anywhere in the value itself.

Additionally, the two `fastcgi_params` settings were merged into one dictionary so that further analysis might be easier. This is possible because their order does not matter. A counter-example would be the `location` setting – the order of these settings is important, both among themselves, and relative to other settings, so they cannot be grouped into a dictionary.

7 Conclusion

In this work, we analyzed data structures used in system configuration files to have a foundation for the semantic analysis of their contents in our future research in the file of automated configuration management. We found that even custom configuration formats can usually be expressed using one of several commonly used data structures. However, some software packages are configured in a Turing-complete programming language, usually the same one that the configured program was written in, and there are therefore unsuitable for our analysis. Regardless, we provided a taxonomy of configuration formats, and highlight common themes in custom formats used by various system software packages. Ultimately, we describe a data structure that is able to hold information about all of these configuration files for further analysis. Our analysis has been applied to configuration files available in the Debian Linux distribution, described classes of errors that may arise in them, and defined a data structure that can contain data in these configuration files in a common format, suitable for later analysis.

References

1. Rabkin, A., Katz, R.: Static extraction of program configuration options. In: Proceedings of the 33th International Conference on Software Engineering (2011)
2. Xu, T., Zhang, J., Huang, P., Zheng, J., Sheng, T., Yuan, D., Pasupathy, S.: Do not blame users for misconfigurations. In: Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles, pp. 244–259. ACM (2013)
3. Xu, T., et al.: Hey, you have given me too many knobs!: understanding and dealing with over-designed configuration in system software. In: Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering. ACM (2015)
4. Xu, T., et al.: Early detection of configuration errors to reduce failure damage. In: 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 2016). USENIX Association (2016)
5. Shambaugh, R., Weiss, A., Guha, A.: Rehearsal: a configuration verification tool for puppet. In: Proceedings of the 37th ACM SIGPLAN Conference on Programming Language Design and Implementation. ACM (2016)
6. Raab, M., Barany, G.: Challenges in validating FLOSS configuration. In: IFIP International Conference on Open Source Systems. Springer, Cham (2017)
7. Xu, T., Zhou, Y.: Systems approaches to tackling configuration errors: a survey. ACM Comput. Surv. (CSUR) **47**, 4 (2015)
8. Huang, P., Bolosky, W., Sigh, A., Zhou, Y.: KungfuValley: a systematic configuration validation framework for cloud services. In: Proceedings of the 10th ACM European Conference in Computer Systems (2015)
9. Tang, C., et al.: Holistic configuration management at Facebook. In: Proceedings of the 25th Symposium on Operating Systems Principles. ACM (2015)

10. Oppenheimer, D., Ganapathi A., Patterson, D.: Why do Internet services fail, and what can be done about it?. In: USENIX Symposium on Internet Technologies and Systems, vol. 67 (2003)
11. Delaet, T., Joosen, W., Van Brabant, B.: A survey of system configuration tools. In: LISA, vol. 10 (2010)
12. Anderson, P., Scobie, A.: LCFG: the next generation. In: UKUUG Winter Conference (2002)
13. Heap, M.: Ansible. Apress (2016)
14. Hosmer, B.: Getting started with salt stack–the other configuration management system built with python. Linux J. **2012**, 223 (2012)
15. Loope, J.: Managing Infrastructure with Puppet: Configuration Management at Scale. O'Reilly Media Inc, Sebastopol (2011)
16. Burgess, M.: Cfengine: a site configuration engine. In: USENIX Computing systems (1995)
17. Święcicki, B.: A novel approach to automating operating system configuration management. In: Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology–ISAT 2015–Part II. Springer International Publishing (2016)
18. Xu, T., et al.: How do system administrators resolve access-denied issues in the real world? In: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems. ACM (2017)
19. Huang, Q., et al.: Identifying self-admitted technical debt in open source projects using text mining. In: Empirical Software Engineering (2017)

Risk-Based Decision Making in IoT Systems

Grzegorz J. Blinowski(✉)

Institute of Computer Science, Warsaw University of Technology,
Nowowiejska 15/19, 00-665 Warsaw, Poland
g.blinowski@ii.pw.edu.pl

Abstract. We propose a general framework for systems of the Internet of Things based on ideas of reliability trust, transaction gain, and risk. The major contribution of this work is the integration of the reliability trust estimated at the perception layer and risk estimation based on global control information provided by the cloud layer. We propose establishing a decision-making process on a random variable-based gain-loss model and methods of choice under risk. The proposed approach is general – it can be used for a wide range of trust-based decisions undertaken in the perception layer, such as routing, sensor selection, data aggregation, intrusion detection, and others.

Keywords: Internet of things · Wireless sensor networks · Security · Trust management · Risk-based decision making

1 Introduction and Background

1.1 IoT Architecture – Outline

The Internet of Things (IoT) is a communication system paradigm in which the objects of everyday life that are equipped with microcontrollers, wireless or wired transmitters, and suitable protocol stacks that allow them communicate with one another and, via ubiquitous cloud, also with users, become an integral part of the Internet environment [1].

Here, we will consider an IoT model consisting of three major levels:

- "Perception" (or "things") layer, which encompasses a wide range of "smart" devices ranging from RFID and NFC enabled tags, environmental sensors and actuators, home appliances and mobile terminals.
- "Core" (or network) layer providing heterogeneous communication infrastructure based on multiple network standards including well established L1/L2 technologies such as Wi-Fi, 3G, and Ethernet together with the standard internet protocol suite (IPv4/IPv6 and transport layer UDP/TCP stack) as well as relatively new standards such as 4G LTE, Z-wave, ZigBee, 6LoWPAN, VLC, mIP [2], and others.
- "Cloud" (or application) layer for integrating, managing and analyzing IoT devices. The cloud not only gathers data and manages the "things" and "core" layer, but acts as a ubiquitous service provider for end-users, according to the Service Oriented Approach (SOA) paradigm.

The above IoT model is compatible with the reference architecture model proposed by the EU FP7 IoT-A project [3] and the IoT-A tree structure [4].

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 38th International Conference on Information Systems Architecture and Technology – ISAT 2017*,
Advances in Intelligent Systems and Computing 655, DOI 10.1007/978-3-319-67220-5_21

1.2 IoT Applications

IoT is widely, although mostly anecdotally, known as a network of household appliances – from PC equipment and peripherals to fridges, coffee machines, etc. However, the scope of IoT deployments is much wider, and covers the following areas [1, 5–7]:

- "Smart Cities" – structural health monitoring, noise mapping, traffic congestion monitoring and "smart roads"; smart lighting; waste; security and physical intrusion detection.
- Smart agriculture and farming – fertilizer, pesticide and irrigation monitoring, crop level monitoring; hydroponic plant monitoring and control; animal tracking.
- "Smart environment" – weather monitoring; disaster early warning systems, e.g. flood detection and volcano monitoring); water quality monitoring; chemical leakage and pollution level detection.
- "Smart Grid" – energy consumption monitoring and management; monitoring and optimizing performance in small green energy plants (e.g. solar or bio-gas).
- Industrial Security and Emergency –gas level and leakage detection in industrial environments, radiation level measurement.
- Retail and logistics – supply chain control, NFC payment, intelligent shopping, product management; item location; fleet tracking.
- eHealth – patient surveillance in medical facilities; assistance for elderly or disabled people living independently.
- Home automation ("Smart homes") – energy and water use monitoring, remotely controlled appliances, intrusion detection systems.

It is worth noting that in many of the above areas, especially regarding environmental and agricultural monitoring, industrial security and smart city applications, large networks of power- and resource-constrained wireless sensors are used. This type of IoT infrastructure will be our major focus in this work.

1.3 Trust in IoT

The two traditional methods of protecting IT systems, including IoT environments, are cryptography and access control. They can be considered to be "hard" security measures, which guarantee system security – if implemented correctly. However, in heterogeneous IoT environments, cryptography alone cannot guarantee security, as compromised network nodes can generate false or misleading information while still providing valid cryptographic credentials. Similarly, access control mechanisms are not immune to internal malicious attacks, and traditional centralized access control is not suitable for distributed environments. However, trust management, considered to be a "soft" security measure, can resolve the above-mentioned issues, not necessarily by substituting, but rather by augmenting the "hard" security measures.

The difference between "hard" and "soft" security measures was first described by Rasmussen and Jansson [8], where the term "hard security" was used for traditional mechanisms like authentication and access control, and "soft security" for "social control" mechanisms in general – of which trust is an example. The idea of trust is a central concept in decision-making processes, especially those decisions involving access control schemes, security policy execution, and networking (e.g. routing, data aggregation, localization, node selection, attack detection, etc.). Despite being seen as of paramount importance, trust is a fuzzy concept – no single definition of it exists, and it is defined, calculated and used in various and often contradicting ways [9].

The concept of trust in computer science emerges from the sociological, psychological and economical environments. The origins of computational trust date back to the 90s, when Marsh [10] analyzed its basic properties and replicated social and psychological factors of trust in a computational setting. Since then, numerous trust management systems for different applications and environments have been developed. Trust has been investigated heavily in various networking environments, namely Peer-to-Peer (P2P) networks [11], Wireless Sensor Networks (WSNs) [12], Mobile ad-Hoc Networks (MAHNs) [13] and Grid Computing systems [14]. Surprisingly, up until now, there has been very little effort made to tackle the issues of trust management specifically in IoT environments [15] and even less attention devoted to analyzing risk in distributed systems. In this work, we will focus on the "soft" reputation-based trust model[1] that relies on the history of interactions between peers, as it is agreed that in mid- and large-scale distributed systems, reputation-based trust or a combination of reputation-based and policy-based trust is the right choice [9].

A natural evolution path for researching trust in IoT is to extend the work conducted for P2P systems and WSNs [12]. However, developing a trust framework for the IoT environment is more complex than in the case of P2P or WSNs, as these systems are rather more homogenous than the IoT environment. In WSNs, all nodes share identical or similar architecture and communicate on the same network abstraction level. More complex WSN systems are based on the "sensor, router, sink" paradigm, but they still have simpler architecture and are smaller in scale than IoT systems. In P2P systems, the network consists of identical nodes, or in some cases, two distinct groups of nodes operating on the same resources and based on the same application-level protocol. The case is very different for IoT – there is a fundamental difference in resource availability, architecture and protocols used between the perception, core and cloud layers. The major problem with establishing a trust model in this environment comes down to the issue of propagating trust information and trust control across the three IoT layers. This work presents a model that aims to address this issue.

1.4 Scope of This Work

In this work, we will propose a general framework for IoT systems which is based on ideas of reliability trust, transaction gain, and risk. The major contribution of our approach is the integration of the reliability trust estimated at the perception layer and risk estimation based on global control information provided by the cloud layer. The approach is general – it can be used for a wide range of trust-based decisions undertaken in the perception layer.

[1] An alternative is *policy-based trust*, which is based on predefined rules and credentials, and is in fact a ``hard' security measure.

Yan, Zhang and Vasilakos [16] define ten trust objectives that should be accounted for in a trustworthy IoT system. The scope proposed by the authors is broad and ranges from data transmission trust to issues relating to human-computer trust interaction. From these objectives, we have selected the following which are addressed by our risk-based model:

- Data perception trust (DPT) – concerns mainly the perception layer, and includes properties like sensor sensibility, preciseness, security, reliability, and persistence.
- Trust relationship and decision (TRD) – provides an effective way to evaluate trust relationships of the IoT entities in order for them to securely communicate and collaborate with each other.
- Data transmission and communication trust (DTCT) – related to the security and privacy properties of an IoT system in which light security/trust/privacy is needed. Trusted routing and key management are two important issues required to achieve this objective.
- Quality of IoT services (QIoTS) – implies that the IoT services should be personalized and offered at exactly the right place and time to the right person. This objective is mainly about the trust management in the application layer, but requires support from the other layers.
- Identity trust (IT) – concerns the objective properties, e.g. identity privacy, and subjective properties, e.g. user hope.

The paper is structured as follows: in Sect. 2 we discuss the notion of trust in the IoT perception layer and in IoT systems in general. In Sect. 3, we describe the proposed model and discuss its properties. Section 4 summarizes the presented work.

2 Trust and Risk in Sensor Networks and the IoT Environment

A full discussion of all the aspects of trust in IoT systems is beyond the scope of this paper, so we refer the reader to recent surveys on this subject – [16, 17]. Here we would like to discuss the most important aspects of trust in IoT. We commence with trust management issues in WSN and MAHN, and extend these issues to the cloud layer of the IoT system. We shall start with a definition of reliability trust characterized in the "transactional" way (after [18, 19]):

Definition 1. ***Reliability trust*** is defined as the subjective probability estimate $\boldsymbol{p}$, by which the trustor expects the trustee to successfully execute the transaction commissioned by the trustor.

The above definition is commonly accepted when modelling trust in distributed systems, because it captures the essential issue of using trust in the decision-making process: to execute a remote transaction, i.e. to delegate some action to another network node (trustee), the trustor must subjectively estimate the probability of success. The transaction's scope can be broad, e.g. (1) routing both atomic packet forwarding decisions and more complex actions (such as accepting a forwarding rule or forwarding table from a peer node); (2) sensor (group) selection (to carry out a specific task or provide specific data); (3) data aggregation (data is aggregated "on the fly" as it moves through the sensor plane towards the core and cloud); (4) authentication and key management (a trust-based approach to key management protocols is an alternative to deploying a hierarchical PKI-based key management system in distributed large scale systems, as this is problematic [20]. In trust-based key management systems, transactions relate to building security associations between peers [21] and acceptance of locally issued certificates [22, 23]); (5) intrusion detection (hostile – compromised - nodes may try to disrupt the system. In Local Intrusion Detection Systems [LIDS], intrusion detection is performed among trustworthy participating nodes); (6) access control (trust is used to determine whether or not to grant access to certain resources or rights).

The second important aspect of trust given in Definition 1 is its "*subjectivity*". Reliability trust is derived from a combination of experience, i.e. 1-to-1 interactions (direct trust) and received referrals (indirect trust). Hence, the trust in the trustee to behave in a given way is generally aggregated from different sub-trust factors: direct and indirect *functional trust* (relating to trusting the given party to execute a transaction), *referral trust* (trust in providing reliable reputation information, i.e. trust estimation for other system nodes), and *identity trust* (a measure of the correctness of a claimed identity over a communication channel). Because of the subjectivity of trust, there is a difference between a trustor's beliefs (reliability trust) and the actual *trustworthiness*[2] of the trustee. This discrepancy leads to *risk*. Risk always arises in the decision-making process; the trustor's goal is to minimize it in a given transaction.

Many schemes for estimating trust have been proposed for WSNs, MAHNs and IoT systems. In general, trust computation in such distributed systems is implemented by estimating the current trust level on the basis of transactional history. Various approaches for this include a simple weighted approach with time fading coefficients and fuzzy logic, Bayesian models, Dempster-Schafer belief theory and belief combining via beta distribution are also commonly used. A useful tool for combining direct, indirect, functional and referral trust is subjective logic formalism [24].

In the overwhelming majority of cases, authors of various trust models or trust-based security schemes analyze their design under two factors: resilience to various attacks specific to the given transaction type, and communication efficiency. This leads to highly specialized algorithms which satisfy performance and security measures but only in their very narrow role. For example, in the recent survey [20], Cho and Swami enumerate over 20 different trust-based routing algorithms for MAHNs. In the next chapter, we will present a more general framework for a trust-based decision-making scheme in the IoT.

3 Model Description

In the perception layer, we will define the set of actions available to each perception-layer node as X. We will consider two mutually exclusive actions: to engage or not to engage in a transaction with a given principal: $X = \{x_A, x_N\}$. S is the set

[2] Trustworthiness is usually defined as the objective probability that the trustee will perform a particular action on which the interests of the trustor depend.

of objective states which indicate whether the given principal was trustworthy or not (i.e. whether the principal was willing to execute the transaction as the trustor expected or not): $S = \{s_T,\ s_F\}$. The consequence function maps all possible combinations of actions and states to a real value: $c(x, s) : X \times S \rightarrow \mathbb{R}$. The probability function *p(s)* expresses the trustor's beliefs, i.e. reliability trust, and reflects the outcome of the estimation of success of a given transaction from the node's trust computing subsystem. The utility function maps consequence to utility, or gain. Its arguments are: consequence c and state $\varsigma : u_\varsigma(c) : \mathbb{R} \rightarrow \mathbb{R}$. ς is taken from the set of all possible node states, which may be represented by a single value or a vector of values. In order not to lose generality, we will not assume a particular domain for a node's state. We will name the outcome of the utility function as "gain", reflecting the fact that the purpose of the transaction execution is to achieve some measurable gain. In the case of failure, gain will be negative.

In economics, utility is a measure of satisfaction experienced by the consumer of a good or service [25]. Utility functions are often used in distributed systems for optimization of parameters such as energy, transmission rate, QoS, etc. Here, the usage of the utility function reflects the fact that the positive or negative outcome of a given transaction as returned by the consequence function should be considered in the context of both the local node state and the global system state. An example of a local state parameter may be available energy: the utility of an energy-consuming operation, such as communication, diminishes as energy resources are depleted. The shape of the utility function determines risk attitudes. For example, the agent would be risk averse if for positive gain, u was a concave function, meaning that, at a particular moment, utility gain from a certain transaction outcome is less that the actual value of the same outcome. The application of utility to trust and risk was proposed in [19], and an example of network optimization via utility functions may be found in [26].

Utility will be our primary measure; hence we will use following gain values:

- g_S – gain from successful transaction execution by peer,
- g_F – negative gain, or loss, when the transaction was commenced but not executed by peer,
- g_N – negative gain, or loss, when trustor abstained from commencing the transaction that the trustee was willing to execute.

Note that the following must hold:

$$g_S > 0 \text{ and } g_F < g_N < 0 \tag{1}$$

Table 1 shows the mapping of action-state combinations to utility (gain). Taking into account the a-priori probability p (as computed by the trust estimation subsystem), we can calculate the expected gain of engaging in the transaction.

The expected loss (negative gain) of abstaining from transactions should also be considered: $(1 - p) * g_N$. This reflects a situation where the decision is made not to trust the peer, but the peer should be trusted, hence we experience a loss of abstaining from action.

Table 1. Utility and risk for all action & state combinations; x represents the action, s the state. Gain is shown in the left table. Risk (gain multiplied by the probability) is shown in the right table.

x / s	x_A	x_N
s_T	g_S	g_N
s_F	g_F	*0*

x / s	x_A	x_N
s_T	$p * g_S$	$(1-p) * g_N$
s_F	$(1-p) * g_F$	*0*

When should a decision to go ahead with a transaction be taken? Obviously, when the expected gain exceeds expected loses:

$$p * g_S + (1-p) * g_F > (1-p)g_N \tag{2}$$

In a simple (and too simplistic) case, we are able to calculate the gain directly, and depending on the value of expected gain g being larger or smaller then 0, decide whether the transaction should be commenced or not:

$$g = p * g_S + (1-p) * g_F - (1-p)g_N \tag{3}$$

In reality, we should rather assume that gain values are represented by random variables represented by their relevant distributions: G_S – gain from successful transaction execution by peer[3], G_F – loss when the transaction was commenced but not executed by peer; G_N – loss when the trustor abstained from commencing the transaction that the trustee was willing to execute:

$$G = p * G_S + (1-p) * G_F - (1-p)G_N \tag{4}$$

In such a case, the decision whether to commence or abstain from a given transaction depends on the evaluation of random variable G (6) in the context of the distribution of intrinsic gain random variables. If probability density functions (pdfs) of G_S, G_F, G_N are known, we can calculate the pdf of G as given in (4) and establish the risk accordingly:

Transaction Risk is evaluated on the basis of the gain (or loss) represented by a random variable G (4).

Figure 1 summarizes the idea of a transaction gain calculation. Values of gain g_S, g_F, g_N and gain distributions G_S, G_F, G_N are illustrated. A normal distribution is used in this example. Reliability trust has a value of p, scalar gain value is indicated as g and combined gain distribution as G. This example shows that when gain is modeled by random variables, there may be a non-zero probability of negative gain (loss), even if a simplified gain calculation shows positive gain. Hence, risk calculation must be employed in the final decision stage.

[3] We will denote random variables with capital letters. $F_G(g)$ is the cumulative distribution function of G.

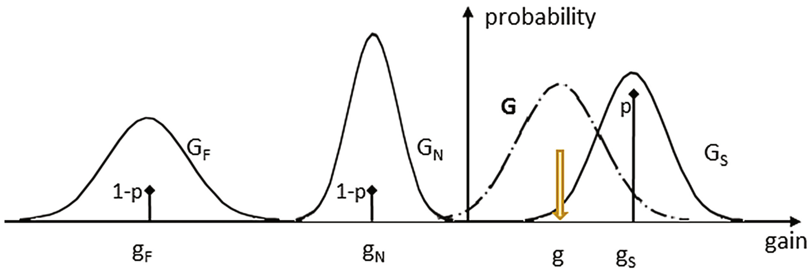


Fig. 1. Example of risk calculation from elementary gains/gain distributions and reliability trust.

An important issue concerns the types of gain distributions that should be used in practical applications. From an analytical point of view, a normal distribution would be the best choice since it provides simple formulas for calculations of its functions. However, this would be too idealistic an approach. Real world models call for fat-tailed and negatively skewed distributions. If cumulative distribution function $F_G(g)$ of G can be calculated, then the decision to trust the peer can be made simply by calculating $F_G(0)$, which will give the probability of generating negative gain (loss) in a given case. The decision can be made on a simple probability threshold, e.g.: "We are willing to accept a 5% chance of loss".

More complex measures based on known risk quantification methods may be applied. The general theory of choice under risk stems from financial mathematics, where risk measures are used to determine the amount of an asset (e.g. money) to be kept in reserve. Because of the subjective nature of risk perception, defining an appropriate measure is controversial and many approaches have been proposed, each one with its own advantages and limitations. Using [27] as a guide, we define risk measure ρ as a mapping between a set of random variables and real numbers: $\rho : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X}$ is a prospect space (a linear space of random variables). The risk measure should satisfy the postulates of (1) Normalization: $\rho(0) = 0$; (2) Monotonicity: if $X \le Y$ almost surely then $\rho(X) = \rho(Y)$; (3) Translational invariance: $\rho(X + m) = \rho(X) + m$ for all $m \in \mathbb{R}$. We refer the reader to the work cited above for a more rigid mathematical formulation.

Expected value and standard deviation risk estimation: this risk quantification is expressed as: $\rho_\gamma^{var} = \mu(G) + \gamma\delta^2(G)$, where $\mu(G)$ is the mean of G, $\delta^2(G)$ is the variance and $\gamma > 0$ is some allowed-spread scaling constant. Alternatively, standard deviation may be used instead of variance: $\rho_\gamma^{var} = \mu(G) + \gamma\delta(G)$. It should be noted that the measures defined above may fail to satisfy the monotonicity requirement. We refer the reader to [28], where risk measures that combine the expected value and the standard semi-deviations are discussed in detail.

Value at Risk (VaR) is a popular financial measure which shows the size of loss under given probability. It is defined as $\mathrm{VaR}_\alpha(G) = \inf\{g \in \mathbb{R} :\ 1 - F_G(-g) \ge \alpha\}$ This definition is equivalent to $Var_\alpha(G)$ being the negative α-quantile of G.

In order to evaluate risk associated with G, we must know what the distributions of intrinsic variables G_S, G_F, G_N are, and calculate G accordingly (or at least its mean and standard deviation). In general this is not possible at the nodes of the perception layer because of two reasons: (1) Local perception nodes do not have the data necessary to quantify intrinsic random variables; (2) Even if the intrinsic gain distributions are known, in most cases the function *G* cannot be calculated analytically, and numeric methods must be used, but perception layer nodes lack the computational resources for this task.

The calculation of *G* and associated risk may be delegated to the cloud layer. In such a case, risk should be evaluated for each executed transaction, but it may be determined with the use of lookup tables precomputed at the cloud level and fed to the sensor layer on a time-to-time basis. This solution is viable for the following reasons: (1) *G* is parametrized with both the local node state (e.g. available energy level), and possibly some global variables (e.g. a measure of data quality or importance); (2) The cloud layer possesses all the data necessary to determine the parameters of the gain distributions since it integrates data from the perception layer and is aware of the state of the system; (3) The cloud layer possesses the necessary resources to carry out all the computation (numerically, if needed); (4) Since the decision about the transaction is binary, pre-calculated lookup tables will be compact, e.g. assuming 100 intervals for *p*, local node energy level and a global parameter, the lookup table will be 1 million bits, or 122 kB in size without compression; (5) The risk calculation can be universal, i.e. broadcasted to all sensor nodes, but also individually adjusted to particular sensor's needs. Figure 2 illustrates the general communication scheme for risk estimation: reputation is

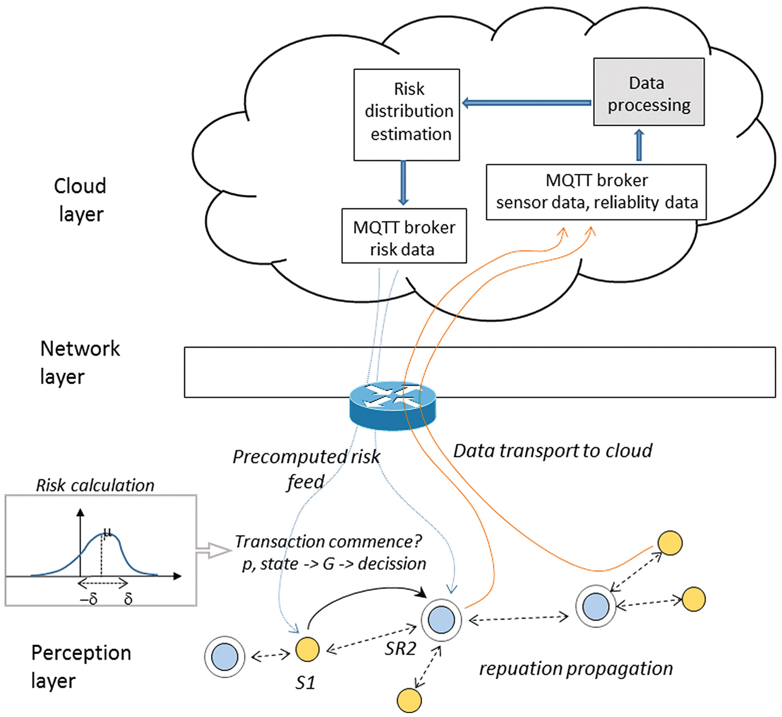


Fig. 2. Communication architecture.

established in the perception layer, trust data is propagated towards the cloud, and risk lookup data is distributed to the perception layer. In the example, sensor node *S1* must determine whether to commence a transaction with sensor-router node *SR2*. With reputation data calculated from its own transaction history and the information received from the perception network (dashed lines), it computes reliability trust p. Next, it uses precomputed gain and risk estimation obtained from the broker to make the decision. The perception network sends data to the cloud layer via sensor-router nodes (orange lines). This data is used in the cloud layer to extract gain/risk related information, which in turn is fed back to the perception layer via data brokers (blue lines).

An example of IoT tailored protocol that suits the needs of risk-related communication is Message Queue Telemetry Transport (MQTT) [29]. MQTT utilizes the publish/subscribe pattern to provide transition flexibility and simplicity of implementation. Its variant MQTT-SN [30] was defined specifically for sensor networks and is based on UDP transport. MQTT can be used with our model for two-way communication: from perception to cloud layer, via brokers with which sensors register; and in the opposite direction: from the cloud to the sensors, which in this case subscribe to risk-based information brokers.

4 Summary

We have proposed a risk-based decision-making framework for the perception layer of IoT systems. Its major feature is that local decision making (done by the sensor node) is based on risk estimates precalculated at the cloud level. This approach is suitable for large-scale perception layer systems, since it keeps the trust-based “fine-grained” decision purely local (i.e. it is based on reliability trust estimated by the reputation system). On the other hand, the risk information used to measure the feasibility of transactions in a given global context is provided by the cloud level.

The framework that we have described in this work is general, and important details need to be worked out in practical implementation. These details include protocols and data formats for delivering risk information to the sensor layer, methods for integrating trust information in the cloud layer, and models for gain distributions and risk calculations. This is the scope of further work on our model. Finally, it is worth mentioning that a logical extension of our model is an evaluation of multiple trustees in order to choose the one with the lowest risk value.

References

1. Atzori, L., Iera, A., Morabito, G.: The internet of things: a survey. Comput. Netw. **54**(15), 2787–2805 (2010)
2. IETF RFC Homepage, IP Mobility Support for IPv4, Revised https://tools.ietf.org/html/rfc5944. Last Accessed 10 May 2017
3. The 7th Framework Programme funded European Research and Technological Development from 2007 until 2013; Internet of Things and Future Internet Enterprise Systems. http://cordis.europa.eu/fp7/ict/enet/projects_en.html. Last Accessed 10 May 2017

4. Architectural Reference Model for the IoT – (ARM). Introduction booklet. http://iotforum.org/wp-content/uploads/2014/09/120613-IoT-A-ARM-Book-Introduction-v7.pdf. Last Accessed 10 May 2017
5. Da Xu, L., He, W., Li, S.: Internet of things in industries: a survey. IEEE Trans. Industr. Inf. **10**(4), 2233–2243 (2014)
6. Jalali, R., El-Khatib, K., McGregor, C.: Smart city architecture for community level services through the internet of things. In: 18th International Conference on Intelligence in Next Generation Networks (ICIN), pp. 108–113 (2015)
7. Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., Ayyash, M.: Internet of things: a survey on enabling technologies, protocols, and applications. IEEE Commun. Surv. Tutor. **17**(4), 2347–2376 (2015)
8. Rasmusson, L., Janssen, S.: Simulated social control for secure internet commerce. In: Proceedings of the 1996 New Security Paradigms Workshop. ACM (1996)
9. Yan, Z., Holtmanns, S.: Trust modeling and management: from social trust to digital trust. IGI Global, 290–323 (2008)
10. Marsh, S.P.: Formalising trust as a computational concept (1994)
11. Gupta, M., Judge, P., Ammar, M.: A reputation system for peer-to-peer networks. In: Proceedings of the 13th International Workshop on Network and Operating Systems Support for Digital Audio and Video, pp. 144–152 (2003)
12. Han, G., Jiang, J., Shu, L., Niu, J., Chao, H.C.: Management and applications of trust in wireless sensor networks: a survey. J. Comput. Syst. Sci. **80**(3), 602–617 (2014)
13. Cho, J.H., Swami, A., Chen, R.: A survey on trust management for mobile ad hoc networks. IEEE Commun. Surv. Tutor. **13**(4), 562–583 (2011)
14. Azzedin, F., Maheswaran, M.: Evolving and managing trust in grid computing systems. In: IEEE CCECE 2002 Canadian Conference on Electrical and Computer Engineering, vol. 3, pp. 1424–1429 (2002)
15. Fernandez-Gago, C., Moyano, F., Lopez, J.: Modelling trust dynamics in the Internet of Things. Inf. Sci. **396**, 72–82 (2017)
16. Yan, Z., Zhang, P., Vasilakos, A.V.: A survey on trust management for internet of things. J. Netw. Comput. Appl. **42**, 120–134 (2014)
17. Lize, G., Jingpei, W., Bin, S.: Trust management mechanism for internet of things. China Commun. **11**(2), 148–156 (2014)
18. Gambetta, D.: Can we trust trust. In: Trust: Making and Breaking Cooperative Relations, vol. 13, pp. 213–237 (2000)
19. Jøsang, A., Ismail, R., Boyd, C.: A survey of trust and reputation systems for online service provision. Decis. Support Syst. **43**(2), 618–644 (2007)
20. Cho, J.H., Swami, A., Chen, R.: A survey on trust management for mobile ad hoc networks. IEEE Commun. Surv. Tutor. **13**(4), 562–583 (2011)
21. Hadjichristofi, G.C., Adams, W.J., Davis, N.J.: A framework for key management in mobile ad hoc networks. In: International Conference on Information Technology: Coding and Computing, ITCC 2005, vol. 2, pp. 568–573 (2005)
22. Virendra, M., Jadliwala, M., Chandrasekaran, M., Upadhyaya, S.: Quantifying trust in mobile ad-hoc networks. In: International Conference on Integration of Knowledge Intensive Multi-Agent Systems, pp. 65–70 (2005)
23. Li, R., Li, J., Liu, P., Chen, H.H.: On-demand public-key management for mobile ad hoc networks. Wirel. Commun. Mobile Comput. **6**(3), 295–306 (2006)
24. Jøsang, A.: A logic for uncertain probabilities. Int. J. Uncertain. Fuzziness Knowl.-Based Syst. **9**(03), 279–311 (2001)
25. Nicholson, W., Snyder, C.: Microeconomic Theory: Basic Principles and Extensions. Nelson Education (2011)

26. Jozefczyk, J., Gasior, D.: Utility-based models and decision making problems for selected network processes. Kybernetes **44**(6/7), 1107–1121 (2015)
27. Artzner, P., Delbaen, F., Eber, J.M., Heath, D.: Coherent measures of risk. Math. Finance **9** (3), 203–228 (1999)
28. Ogryczak, W., Ruszczyński, A.: From stochastic dominance to mean-risk models: semideviations as risk measures. Europ. J. Oper. Res. **116**(1), 33–50 (1999)
29. Locke, D.: MQ telemetry transport (MQTT) v3. 1 protocol specification. IBM developerWorks, Markham, ON, Canada, Tech. Lib https://www.ibm.com/developerworks/library/ws-mqtt/. Last Accessed 10 May 2017
30. Hunkeler, U., Truong, H.L., Stanford-Clark, A.: MQTT-S—A publish/subscribe protocol for wireless sensor networks. In: Proceedings of 3rd International Conference on COMSWARE, pp. 791–798 (2008)

On Implementation of Energy-Aware MPTCP Scheduler

Michał Morawski and Przemysław Ignaciuk(✉)

Institute of Information Technology, Lodz University of Technology, Lodz, Poland
{michal.morawski,przemyslaw.ignaciuk}@p.lodz.pl

Abstract. The majority of networking devices currently available are equipped with a few communication interfaces. However, in most cases, only one of them is used for data transfer as dictated by the design premises of the fundamental transport protocol – TCP. Recently, a variant of TCP – Multipath TCP (MPTCP) – that allows for simultaneous transmission over different paths has been developed. While it allows for spreading the data stream among different interfaces and paths, the way it is actually performed has been left for further investigation. The stream splitting can be optimized according to different quality indicators. The paper discusses the implementation properties of an energy-aware load balancing method tested in a physical networking environment. A comparison with the reference solution is also provided.

Keywords: Multipath TCP · Load balancing · Energy efficiency

1 Introduction

Since the beginning of Internet era the bulk of data exchanged among the communicating devices is subject to the control of the TCP protocol. Over the years, TCP has evolved, adjusting to user demands. However, the majority of its variations concentrate on the congestion control aspects. The communication path is fixed for the duration of connection. If due to any reason (routing events, channel fading, noise, etc.) the applied path is deteriorating in its transmission capabilities, the connection is maintained until the sender interface eventually fails. Even worse – if the path breaks outside the local network, then the TCP connection will be stalled. Such an approach is hardly justified nowadays when the devices are equipped with a few network interfaces (NICs), e.g., Ethernet, Wi-Fi, or cellular, that can be used to shape the transmission more efficiently. Nevertheless, a static usage policy is typically applied, i.e., until one of the interfaces fails – say Ethernet – neither Wi-Fi, nor LTE is activated for the Internet traffic. Multipath TCP (MPTCP) [1, 2] has the potential to elevate the end-point transfer efficiency, and thus the user application performance, by employing simultaneously multiple paths between the communicating parties. Even though the fundamental framework of MPTCP has been standardized [3, 4], some research problems remain unsolved. One of those research challenges is the development of methods of MPTCP stream splitting into multiple sub-streams, controlled by the single path (legacy) TCP – SPTCP.

An advantage of MPTCP is its agnostics to both the application and infrastructure particularities. The appropriate changes are applied inside the TCP stack only.

The popularity of smartphones, tablets, mobile computers, and associated increase of their functional capabilities, has outgrown the progress in battery design. Those devices suffer from fast energy depletion which limits the end-user experience. The majority of energy is consumed by the display and coprocessors, offloading devices, and additional cores run to handle the application data. Therefore, in order to improve the energy economy of an Internet connection, one should shorten the time of data transfer and allow the users to switch off these subsystems or reduce the clock speed. Although addressing similar problems as in the current literature [5–8], the solution presented here differs both in the objectives and in the way the traffic is distributed among the interfaces. It specifies the load-balancing operation of the MPTCP scheduler – in the described approach – the way the MPTCP stream is split into sub-streams. The actual volume of data in a particular flow is determined as a solution of optimization problem formulated for a mathematical model of power consumption. In order to achieve the best results the possibility of acquiring additional information from the link layer driver is explicitly taken into account. A similar approach has been presented in [8], however, the design of the scheduler has been restricted to the TCP layer only there. It makes that solution complex and difficult to implement in a real system.

The theoretical background of the presented load-balancing solution has been covered in [10]. Here, the properties of its physical implementation are discussed. The task is performed on the basis of Linux kernel and using real networks, common devices, and ordinary traffic patterns. The theoretical results have promised significant gain in the actual implementation. It is confirmed through numerous experiments involving physical equipment.

2 Problem Statement

The considered architecture is illustrated in Fig. 1. The research objective is to provide a load-balancing algorithm for the traffic scheduler. The scheduler acquires all the information necessary for the algorithm implementation from the NIC drivers and SPTCP states. In the analyzed communication scenario, the user application seeks to send B bits of data to a remote peer. The data is placed in the TCP output buffer held by the local MPTCP controller. The controller opens n sub-channels and distributes the traffic among the corresponding flows that are regulated by the ordinary SPTCP. The SPTCP control is executed separately for each sub-channel. In the standardized MPTCP implementation [4], three schedulers are designed: "roundrobin" that implements an unweighted round-robin algorithm, "redundant" where the data are flooded over all sub-paths, and the "default" where the sub-path with the lowest SRTT (Smoothed Round Trip Time) is selected. The MPTCP design committee recommends the "default" scheduler. It offers overall good performance, but augments jitter (latency variability). If jitter is problematic for a given application [9], then the "redundant"

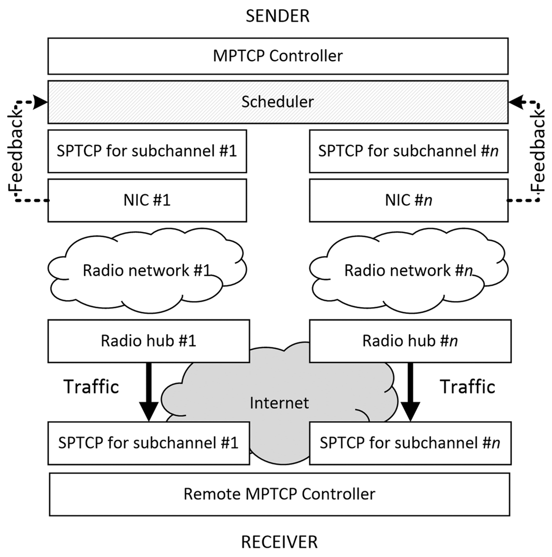


Fig. 1. Assumed MPTCP architecture. The first mile links are likely bottleneck ones.

implementation should be selected. However, it is inconsiderate with respect to the energy expenditure. The solution advocated here joins the benefits of those schedulers while avoiding their major pitfalls.

The scheduler implementation presented in this work leaves the standard behavior of MPTCP (e.g., LIA, OILA) [11] and ordinary TCP [12, 13] intact. The path manager "fullmesh" is always selected, i.e., the number of possible paths is a product of usable interfaces on the client and server side, as detailed in Sects. 4 and 5.

3 Analytical Background

The amount of energy consumed by a networking device depends on many factors. Sometimes handling the NICs prevails, but usually the related activities, e.g., powering the display, GPU, and data processing hardware, dominate. The NIC energy expenditure is related to its type, the transmission power, noise level, number of retransmissions, and control plane activity. These factors differ among interfaces and their mode of operation [10].

Table 1 summarizes the properties of popular interfaces according to [8]. But it is necessary to remember that the actual values may deviate much. The measurements presented in [10] show them change more than twice in a short interval.

The theoretical aspects of the energy-aware scheduling solution chosen here for implementation have been discussed in [10]. For convenience, the major points are

Table 1. Comparison of different MPTCP scheduling methods [8]

	4G	3G	Wi-Fi	Ethernet
Median upload speed [Mbps]	5.64	0.72	0.94	100
Median download speed [Mbps]	12.74	1.10	4.12	100
Download efficiency [mW/Mbps]	52	122.1	137	2
Upload efficiency [mW/Mbps]	438.4	869	238.2	2
Operational power p^{op} [mW]	1288	817.9	132.9	90

recalled below. In order to reduce the transmitting device energy expenditure one needs to optimize E^0 given by

$$E^o = \sum_{i=1}^{n} E_i + P_D \max_i T_i, \ \text{s.t.} T_i > 0, \tag{1}$$

where

$$E_i = \int_0^{T_i} p_i(t)dt > 0 \tag{2}$$

is the total energy dissipated by interface $i \in [1, n]$, $p_i(t) > 0$ is the power necessary to transmit data through interface i at time t, T_i is the total transmission time, and $P_D > 0$ is the power consumed by the system to handle data. $p_i(t)$ [W] is a function of the power efficiency of NIC – $\tilde{p}_i(t)$ [W/bit] and the number of bits sent through channel i – b_i:

$$p_i(t) = \tilde{p}_i(t)b_i + p_i^{op}, \tag{3}$$

where p_i^{op} [W] is the power necessary to keep the NIC in the operational state. $\tilde{p}_i(t)$ and p_i^{op} are highly variable [8, 10]. Their profiles can be obtained using the chipset datasheets but in practice real time measurements are necessary (such an approach is suggested in [5]). The optimal E^0 is achieved when the following condition is satisfied

$$\forall_{i,j \in [1,n]} \frac{b_i}{b_j} = \frac{\tilde{P}_j \bar{c}_i}{\tilde{P}_i \bar{c}_j}, \tag{4}$$

where $\tilde{P}_i = \tilde{p}_i + \tilde{P}_D / n$ – see [10] for derivation details.

4 Implementation

It has been decided to implement the energy-aware scheduler as a new module in Linux kernel 4.3, incorporating the publicly available sources [4]. In this way, the work on the scheduler is conducted in parallel to the development within the main branch of that project, taking advantage of its improvements.

In the discussed implementation, the sessions are established at the beginning of MPTCP transmission using the module "path manager". Path manager decides which sub-paths will be selected for the forthcoming data flows. During the algorithm evaluation, the "fullmesh" path manager was used. It creates SPTCP logical sub-paths that link each NIC of the client with each NIC of the server. Next, the user program fills its buffers and requests to send data. The TCP protocol stack, if the peer is determined to be MPTCP aware, splits the data stream into active sub-flows. The introduced new module is responsible for performing the flow division.

The assumptions taken in the derivation of (4) do not allow one to implement the presented energy-aware method of load balancing directly. First of all, the applied mathematical model neglects the congestion control influence of SPTCP and the scheduler requires the channel capacities and power consumption to be measureable. Secondly, it does not discriminate between short (where the slow-start phase prevails) and long-lived (where the congestion control algorithm is active) connections. An efficient scheduler implementation should take these restrictions into account. It should also be independent of the specifics of SPTCP and application flow continuity.

The analyzed scheduling algorithm assumes $\bar{c}_i$ is the link layer capacity [10], but the existing drivers do not supply such information – it needs to be assessed at a higher level. Besides, the bottleneck can be anywhere on the path – not necessarily at a local link. Therefore, in the discussed implementation, $\bar{c}_i$ is replaced by the TCP-level capacity:

$$\frac{2}{\hat{c}_i} = \begin{cases} \tau_i, & \text{if } \psi = 0, \\ \tau_i/\psi, & \text{otherwise,} \end{cases} \tag{5}$$

where τ_i is SRTT for path i, and ψ_i describes "in-flight" data for path i (the "in flight" data is the data already sent but not yet acknowledged). These values are available in the current SPTCP implementations.

As mentioned above, full implementation of the scheduler requires measuring the energy consumption. Unfortunately, the typical hardware devices available on the market sink about 50% of the max current even if the corresponding driver is not loaded (LTE interface has to be connected via an externally powered USB hub). On the other hand, off-the-shelf drivers lack power monitoring functions, thus enforcing modifications of the associated electronics. Therefore, it has been decided to "mock-up" the energy consumption function and allow for statically assigned (but externally controlled) values. This approach extends the solution delivered in [8], which considered only the median values. Still, to take the full advantage of the proposed energy-aware solution, the hardware should provide such functionalities. In the implementation discussed here, `struct net_device` has been augmented to handle the interface power during upload, download, keeping in operational state statistics.

The following changes relative to the "default" scheduler have been introduced in the `get_subflow_from_selectors` function:

```
-if (tp->srtt_us < min_srtt) {
-    min_srtt = tp->srtt_us;
-    bestsk = sk;
-}
+ in_flight = tcp_packets_in_flight(tp);
+ intf_pwr = get_pwr(tp); // currently mock-up
+ if (0 != in_flight) {
+    curr_quality = intf_pwr * tp->srtt_us / in_flight;
+}
+else {
+    cur_quality = intf_pwr * tp->srtt_us;
+}
+if (cur_quality < min_quality) {
+    min_quality = cur_quality;
+    bestsk = sk;
+}
```

The value `intf_pwr` cannot be too large to avoid overflowing during integer multiplications on 32-bit devices as it might result in difficult to interpret misbehavior.

Note that the proposed changes address also one of the principal drawbacks of the "default" scheduler – the 'sticky' behavior [15]. With the "default" scheduler, even if the approximation of SRTT on the chosen paths is close in value (e.g., 100 ms and 101 ms), the one with the shortest SRTT is used exclusively until the congestion window fills up. The flows are balanced in a long term only. As a result, the jitter in the MPTCP layer increases and larger buffers are necessary. After applying the proposed amendment, the sub-flows are used uniformly in a short term as well. Contrary to the original solution, this beneficial effect is obtained without tricky heuristics.

Short-lived streams

The modification of the "default" scheduler presented above works well for long-lived streams. In turn, the mobile devices, that are the principal consumers of data nowadays, open mostly short-lived sessions [14] (e.g., http requests). For short-lived connections, evaluating the quotient $p_i/\bar{c}_i$ in (4) fails. The existing hardware and device drivers do not allow for acquiring $\bar{c}_i$ directly, and its approximation $\hat{c}_i$ is highly unreliable. However, for short-lived sessions all the data are expedited before the first acknowledgement arrives so SRTT is roughly known. Moreover, establishing a single short-lived session is uncommon. While requesting popular sites, like polish news portal "wp.pl", many other sessions are opened in the same time period (e.g., 25–47 sessions as determined in the laboratory tests we performed 24th Apr. 2017, 14:30-15:00). The actual number of sessions depends on the site content, ad-blockers, or spy-blockers enabled and it cannot be predicted by the kernel. For the upload direction, the majority of sessions are short-lived. They should not be split between the available MPTCP paths.

As a consequence of these observations, it has been decided to change the scheduler behavior only if all the sub-streams are in the slow-start phase and assign subsequent sessions in a static way. When a given interface is selected, its weight is increased by the current power coefficient associated with this interface. In the next round of selection process, the sub-stream is associated (a) with the same interface if pointer to `struct mptcp_cb` is the same as the previous one, and (b) with the interface with lowest weight if a new stream is encountered. After a period of inactivity since the last slow-start phase (in the current implementation 1 s) the selection process is reinitiated. This intentional modification reduces the duration of the series of short-lived sessions thus having positive impact on the "browser" traffic pattern typical for mobile devices [14].

5 Evaluation

In order to test the scheduler implementation the setup presented in Fig. 2 is applied. The popular low-end Raspberry Pi device has been chosen as a client and a high-end virtual machine in the local data center as the server. During all tests the server runs unattended. An influence of developed algorithm on the server is not monitored. To facilitate solution analysis, the client has been equipped with two, while the server with only one interface. As "fullmesh" path manager is selected, exactly two paths are established among the client and the server. Both paths have no common bottlenecks. In our opinion, such system properly approximates a typical usage scenario. The tests were conducted in Apr. 2017, at different times of the day. Except the typical background activities, only *iperf3* program together with *tcpdump* were running on the communicating devices. When SPTCP uses the DSL-path the maximum throughput oscillates around 7.5 Mbps, and for the LTE one – 2-5.5 Mbps. The scheduler properties have been evaluated on the basis of the *pcap* file using the popular *Wireshark* program. It has been decided to test the solution using open Internet instead of a closed laboratory network, as it leads to more practical conclusions (also, in the Internet environment, the measurements are hardly replicable). In order to avoid particular networking conditions, numerous tests have been conducted at multiple days. Each test

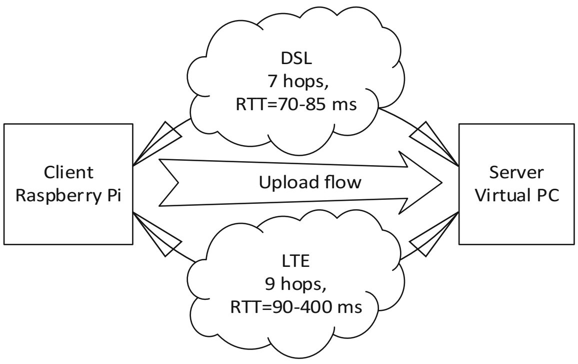


Fig. 2. Test setup. The path properties are detailed in the "clouds".

encompasses a 10 s measurement using the "default" scheduler followed immediately by a 10 s test using the assessed energy-aware one. Different patterns of power consumption have also been evaluated (inter alia leaving or arriving the access point coverage area – linear changes, leaving or entering building, switching BTS – step wise changes, link layer retransmissions – oscillatory changes).

5.1 Long-Lived Streams

Table 2 presents example values of power dissipated by the client device in different networking conditions. For Ethernet (DSL path) it has been assumed 90/7.5 = 12 [mW/Mbs]. Consequently, the interface power dissipation ratio Eth/LTE is about 40 and such value has been selected to validate the implementation. The bias of application-induced power consumption is assumed as $P_D = 600$ [mW]. In the majority of tests one can observe non-negligible energy savings. The algorithm's gain drops if congestion windows are exhausted on some paths. If so, scheduler works in handicapped way – only one path is active. The scheduler cannot place data on sub-stream, which is temporary blocked.

Table 2. Results of experiments: B_D – Throughput [Mbps] of DSL channel, B_L – Throughput [Mbps] of LTE channel, T – Time of transmission [s], E – dissipated energy [J], G – gain in applying energy-aware solution [%]. R – rush hours, W – wee hours.

"default" scheduler				Proposed scheduler				G	When
B_D	B_L	T	E	B_D	B_L	T	E		
7.3	2.0	8.58	5.26	7.4	2.8	7.89	4.85	8	W
7.2	2.2	8.2	5.22	7.1	3.5	7.54	4.65	11	W
6.8	1.8	9.35	5.71	6.7	2.1	9.08	5.55	3	
6.7	1.4	9.82	5.99	6.7	2.9	8.15	5.14	14	
5.5	1.4	11.59	7.04	5.4	3.4	9.09	5.56	21	
3.0	0.7	22.55	12.98	3.1	1.8	16.60	10.02	23	R
2.8	0.7	22.82	13.73	2.7	2.8	14.60	8.83	36	R

The outcomes presented in Table 2 are obtained for fixed power coefficients $\tilde{P}_i$. The coefficients vary with time, however, as a result of natural fluctuations and retransmissions. After having tested different kind of change patterns no significant influence of fast oscillations of $\tilde{P}_i$ on the overall energy consumption has been observed. Only when changes are much longer than SRTT, the effect becomes noticeable. It relaxes potential requirements concerning power measurements.

It has been observed as an opportune side effect that the tested scheduler works better than the "default" one even if power coefficients $\tilde{P}_i$ are all equal. The proposed scheduler allows for sending 0–10% more data in the same time frame as the "default" solution. The gain increases during rush hours, and lowers almost to zero at night, owing to better approximation of the path capabilities. Additionally, a reduction of the number of subsequent segments expedited by the same interface has been obtained. The number decreased from several (sometimes over 30) to 2–4 segments thus reducing jitter. Such behavior is especially important for applications using low-end devices typical for Internet of Things [16, 17]. The discussed positive effect eliminates the necessity of incorporating the "redundant" scheduler in most cases.

Particularly promising results have been obtained for non-persistent streams, i.e., the streams whose throughput is constrained not by the network but by the overlaying application (Internet radio, interactive sessions (vnc, ssh), or video-on-demand). When a session with higher power consumption has a lower SRTT, the power savings in using the energy-aware scheduler instead of the "default" one can be as high as 90%. For such sessions, the congestion window fills up infrequently, so the "default" scheduler favors the path with the smaller SRTT, but not the most "green" one.

5.2 Short-Lived Streams

The short-lived sessions are often neglected during the assessment of TCP variants. However, as pointed out in [14], such sessions prevail over long-lived ones during upload for the typical usage pattern involving mobile devices. A single short-lived stream does not influence the effectiveness (both time and power) significantly. It becomes an issue when the congestion windows are small (owing to background activity, or many short sessions opened at the same time). The proposed solution decreases the time of data transfer (tested using *iperf3* with 10 parallel sessions and 10 segments per session). Nevertheless, essential gain has not been observed (only up to 5–7%) even if hundreds of short-lived sessions are created. Careful investigation of the *tcpdump* log shows that the path manager, i.e., the piece of software responsible for establishing the paths, constitutes the bottleneck. Irrespective of the actual needs, the "fullmesh" version establishes all possible paths. The time of such operation is comparable with the time of transferring the data and is the primary source of perturbation. Therefore, the scheduling solution for short-live sessions should be complemented by a modification of the path manager so that additional sub-streams are opened only when necessary.

6 Summary and Conclusions

The paper discusses practical evaluation of a new scheduling algorithm designed for the MPTCP traffic handling. The scheduler implementation, built on the basis of the "default" one, has been tested in a real networking environment, involving the Internet connectivity and physical devices. As opposed to the original – heuristic – solution, the new one originates from analytical optimization. Numerous tests conducted in a real networking environment show its superiority over the "default" scheduler. The improved efficiency is achieved at negligible computational cost. Its current implementation is suboptimal due to the lack of hardware support but open for an easy extension. Even statically introduced data about the power consumption allows saving approximately 20% of the energy spent by the networking device on serving the transmission. The algorithm responds properly to (artificially injected) fluctuations of energy consumption. Further work on the problem will be focused on improving the energy ratings during short-lived sessions as well as simultaneous control of upload and download streams.

References

1. Peng, Q., Walid, A., Hwang, J., Low, S.H.: Multipath TCP: analysis, design, and implementation. IEEE/ACM Trans. Network. **24**(1), 596–609 (2016)
2. Xu, C., Zhao, J., Muntean, G.: Congestion control design for multipath transport protocols: a survey. IEEE Commun. Surv. Tutorials **18**(4), 2948–2969 (2016)
3. Ford, A., Raiciu, C., Handley, M., Bonaventure, O.: TCP Extensions for Multipath Operation with Multiple Addresses. RFC 6824 (2013)
4. Barré, S., Paasch, C.: MultiPath TCP – Linux Kernel implementation. http://www.multipath-tcp.org. Last accessed 15 May 2017
5. Deng, S., Netravali, R., Sivaraman, A., Balakrishnan, H.: WiFi, LTE, or Both? Measuring multi-homed wireless internet performance. In: Proceedings on ACM IMC, Vancouver, Canada, pp. 181–194 (2014)
6. Paasch, C., Ferlin, S., Alay, O., Bonaventure, O.: Experimental evaluation of multipath TCP schedulers. In: Proceedings on ACM SIGCOMM CSWS, Chicago, USA, pp. 27–32 (2014)
7. Hwang, J., Yoo, J.: Packet scheduling for multipath TCP: In: Proceedings on 7th International Conference on Ubiquitous and Future Networks, Sapporo, Japan, pp. 177–179 (2015)
8. Peng, Q., Walid, A., Chen, M., Low, S.: Energy efficient multipath tcp for mobile devices. In: Proceedings on 15th ACM International Symposium on Mobile Ad Hoc Networking and Computing, Philadelphia, Pennsylvania, USA, pp. 257–266 (2014)
9. Ignaciuk, P., Bartoszewicz, A.: Discrete-time sliding-mode congestion control in multi-source communication networks with time-varying delay. IEEE Trans. Control Syst. Technol. **19**(4), 852–867 (2011)
10. Morawski, M., Ignaciuk, P.: Energy efficient dynamic load balancing in multipath TCP for mobile devices. In: Proceedings on 37th International Conference on Information Systems Architecture (ISAT), Karpacz, Poland, pp. 162–171 (2016)
11. Khalili, R., Gast, N., Popovic, M., Boudec, J.-Y.L.: MPTCP is not pareto-optimal: performance issues and a possible solution. IEEE/ACM Trans. Network. **21**(5), 1651–1665 (2013)
12. Abdeljaouad, I., Rachidi, H., Fernandes, S., Karmouch, A.: Performance analysis of modern TCP variants: a comparison of Cubic, Compound and New Reno. In: Proceedings of the 25th Biennial Symposium on Communications (QBSC), Kingston, USA, pp. 80–83 (2010)
13. Ignaciuk, P., Karbowańczyk, M.: Active queue management with discrete sliding modes in TCP networks. Bull. Pol. Acad. Sci. Tech. Sci. **62**(4), 701–711 (2014)
14. Bonaventure, O., Paasch, C., Detal, G.: Use Cases and Operational Experience with Multipath TCP. RFC 8041 (2017)
15. Arzani, B., Gurney, A., Cheng, S., Guerin, R., Loo, B.: Impact of path characteristics and scheduling policies on MPTCP performance. In: Proceedings on 28th International Conference on Advanced Information Networking and Applications Workshops (AINA), Victoria, BC, Canada, pp. 743–748 (2014)
16. Morawski, M., Ignaciuk, P.: Reducing impact of network induced perturbations in remote control systems. Control Eng. Pract. **55**(10), 127–138 (2016)
17. Morawski, M., Ignaciuk, P.: Network nodes play a game – a routing alternative in multihop ad-hoc environments. Comput. Netw. **122**, 96–104 (2017)

Mobile Monitoring System for Environment Parameters

Gerard Żmuda[1], Andrzej Opaliński[1](✉), and Mirosław Głowacki[1,2]

[1] AGH University of Science and Technology, Al. A. Mickiewicza 30, 30-059 Krakow, Poland
gerardzmuda@wp.pl, andrzej.opalinski@agh.edu.pl, glowacki@metal.agh.edu.pl

[2] The Jan Kochanowski University, Ul. Żeromskiego 5, 25-001 Kielce, Poland

Abstract. Problems related to the monitoring of environmental parameters, and in particular air-pollution in urban agglomerations, are some of the more publicly discussed public issues in recent times. This paper presents the concept, functionality, main hardware and software components, and test results of low-cost prototype of the system, enabling monitoring of environmental parameters such as air-pollution, temperature, humidity, pressure and UV radiation. The main features of this solution are its low cost, relatively high precision of measurement and the ability to operate in mobile version, without wired power supply and communication, including GPS coordinates as one of the measured parameters. The prototype can be an alternative to expensive professional solutions and a base for developing more advanced sensor networks. For home use it can be used as a home weather station with an air pollution sensor, being an extension of the of Personal Area Network of the recently popular Internet of Things solutions.

Keywords: Mobile monitoring system · Air pollution monitoring · Environmental parameters monitoring

1 Introduction

Popularization of wireless communications and miniaturization of computer components has enabled humanity to develop the concept of Internet of Things. It can be described as a combination of miniature devices with the Internet connection, so that they can fulfill their functionalities.

The main sources of air and soil pollution are industry, traffic and urban transport. Therefore, the society of threatened cities starts taking actions to reduce pollution and constant monitoring of pollutant levels in order to avoid contact with contamination in case of its emergence. One of such an air quality control system is located in Cracow It consists of several measuring stations located in the city center and on its outskirts. Data collected from each station are displayed in graphs on the website of the Provincial Inspector of Environmental Protection in Cracow.

These measurement stations, however, are an expensive solution that only large institutions can afford. However, the previously mentioned Internet of Things gives us the opportunity to create a smaller, more accurate and cheaper alternatives that can be applied in home conditions. The prototype of such a device, called "SmogoMetr", has been developed and described in the presented article. The developed device provides the possibility to carry out research in the field of air pollution monitoring, without purchasing expensive measuring infrastructure. Device mobility and wireless connectivity ensure that the station can be moved to any location while maintaining its functionality.

2 Related Works

The issue of monitoring environmental parameters is a science and research field that requires the integration of solutions in the fields of mechanics, electronics, power engineering, computer networks and measurement systems. Such systems are used in a wide range of research, from weather monitoring systems [5], air pollution [12], environmental conditions of animal life [6]. Air-pollution issues are recently very popular not only in Poland but also in the world [8, 12]. Examples of such research are works in Mauritius, concerning creation of a network of air pollution monitoring sensors and a system that aggregates these data into a common database [1].

Issues related to monitoring environmental parameters involve many aspects of engineering such as sensor networks [10], integration of various types of data [7, 13], sensor fabrication [9], calibration [3] and error detection [11] to ensure the quality of the readings returned by those sensors, networks and systems. A separate issue in such systems is to consider them as mobile systems, taking into account the GPS coordinates of their systems [2, 10].

Until recently, such research was conducted on the basis of large expensive installations that only large scientific research centers or government organizations could afford [12]. With the miniaturization of computer components and the popularization of the Internet of Thing solutions, an increasing number of low-cost solutions have emerged, allowing for similar research based on considerably less financial resources [4, 14].

The goal that the authors of this work have put forward is to implement a system that, while maintaining the quality of measurement, will be inexpensive, flexible and will allow easy testing at various locations. Similar systems have been developed in the research previously, but usually covered different ranges of measurement parameters and other hardware architectures of the solution [2–4].

3 Solution Concept and System Architecture

The goal of the project presented in this article was to develop a mobile environment monitoring system, whose main functions are:

- real-time monitoring of selected environmental parameters (dustiness, temperature, pressure, humidity and ultraviolet radiation),
- archiving of monitored parameters,
- mobility of the system: own power supply and combining measurements with GPS parameters,
- graphical representation of monitored parameters (virtual map).

The above functionality is presented in Fig. 1.

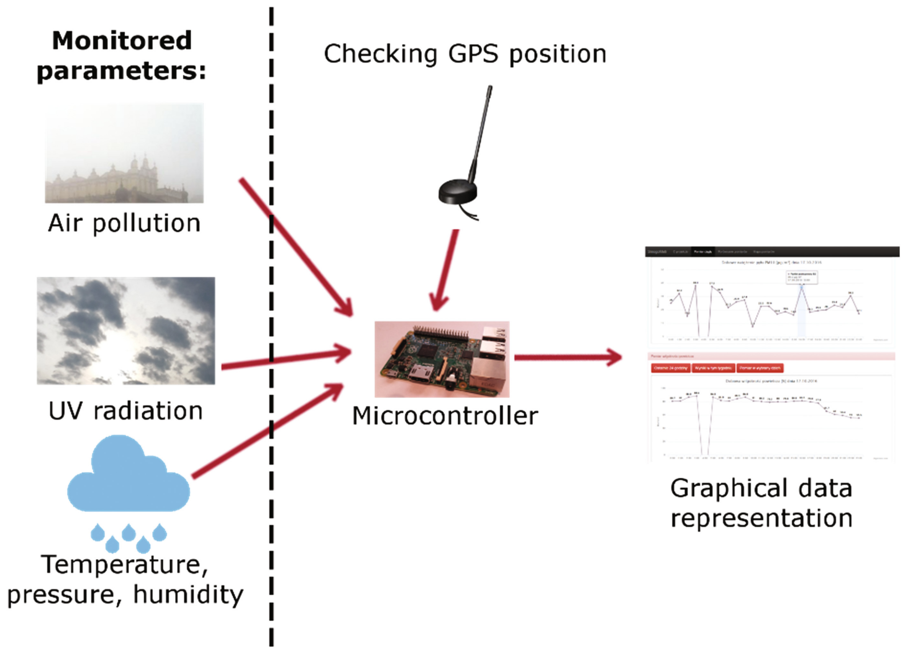


Fig. 1. Main system functionalities.

In order to implement a system providing such functionality, it was necessary to design a system, the main component of which was the Raspberry Pi controller. Hardware-related part of the project was an assembly and configuration of hardware sensors (pollution, pressure, humidity and UV radiation) and GPS and GSM modules. In addition, a graphical user interface has been implemented that enables the presentation of the results collected by the system. The last stages of work on the system were the system calibration phase and analysis of the results obtained with the system.

The development of the software part of the prototype included software for reading sensor parameters and sending the collected data to the server. The server-side application processes the received measurements, places them in the database, and presents them in a graphical user interface, that can be used by system users to check current weather conditions and environmental pollutant parameters. The technologies used to implement the components of the system are shown in Fig. 2.

- SmogoMetr measurement scripts have been implemented using the Python language,
- The server uses PHP to receive a SmogoMetr message and a MySQL database engine to archive the entry,
- Client requests are handled by the server using JSON and PHP scripting language,
- WEB interface was developed using HTML, JavaScript, and JQuery.

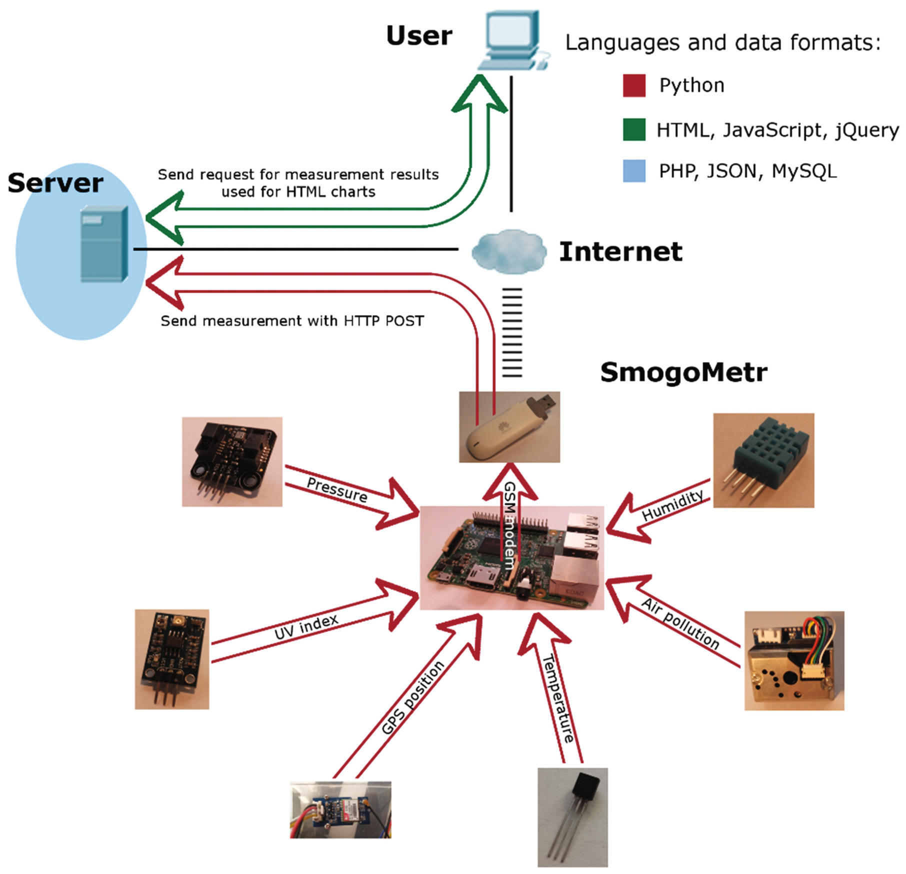

User
Languages and data formats:
🟥 Python
🟩 HTML, JavaScript, jQuery
🟦 PHP, JSON, MySQL
Server
Send request for measurement results used for HTML charts
Internet
Send measurement with HTTP POST
SmogoMetr
Pressure
Humidity
GSM modem
UV index
Air pollution
GPS position
Temperature

Fig. 2. Technologies, protocols and programming languages.

The prototype platform launch measuring scripts every 2 min, resulting a total of 30 measurements per every hour of operation. Each measurement is sent by established a connection via the GSM module to a server, which has a static IP address or domain name. Communication between system components is shown in Fig. 3.

SmogoMetr does not receive any response about result of data processing by the server by default. This action is intended to counter attack by unauthorized use of brute-force algorithms. In addition, the data received from the sensors are full of redundant information that is removed before the measurements are sent. The entire process of extracting the desired information is performed just after receiving the measurements from each sensor.

Calculation of the air-pollution and UV radiation values is based on the arithmetic mean of a series of measurements taken at 1 s for air-pollution and 0.5 s for UV. This results with accuracy at relatively short waiting times. In order to compensate for errors resulting from the occurrence of individual disturbances, a series of 10 measurements is performed, on the basis of which their arithmetic mean is calculated. Measurement

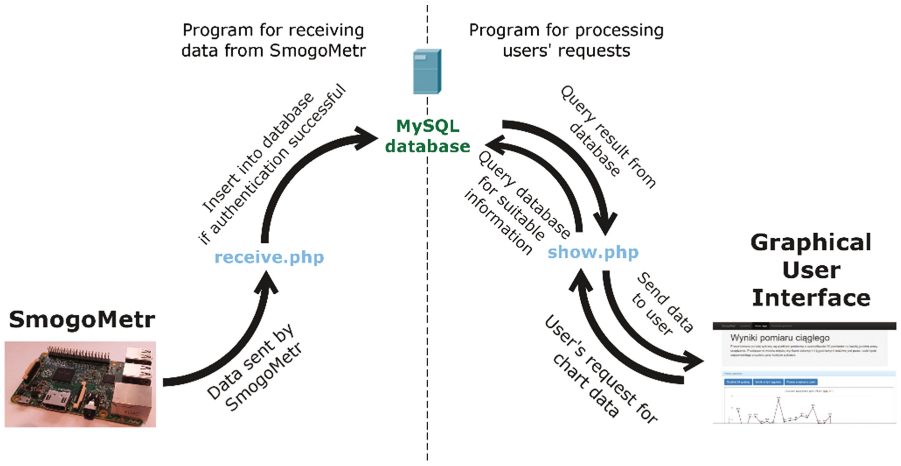
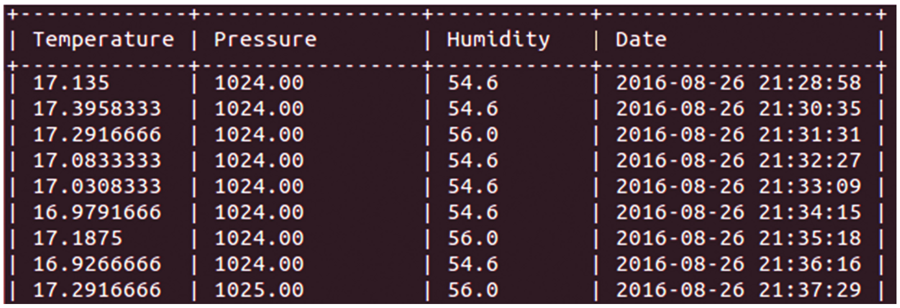

Program for receiving data from SmogoMetr
Program for processing users' requests
MySQL database
Insert into database if authentication successful
receive.php
Query result from database
Query database for suitable information
show.php
Graphical User Interface
SmogoMetr
Data sent by SmogoMetr
Send data to user
User's request for chart data
Wyniki pomiaru ciągłego

Fig. 3. Communication schema.

series are performed at two-minute intervals, which allows for 30 measurements per hour of system operation.



```
+-------------+-----------------+------------+---------------------+
| Temperature | Pressure        | Humidity   | Date                |
+-------------+-----------------+------------+---------------------+
| 17.135      | 1024.00         | 54.6       | 2016-08-26 21:28:58 |
| 17.3958333  | 1024.00         | 54.6       | 2016-08-26 21:30:35 |
| 17.2916666  | 1024.00         | 56.0       | 2016-08-26 21:31:31 |
| 17.0833333  | 1024.00         | 54.6       | 2016-08-26 21:32:27 |
| 17.0308333  | 1024.00         | 54.6       | 2016-08-26 21:33:09 |
| 16.9791666  | 1024.00         | 54.6       | 2016-08-26 21:34:15 |
| 17.1875     | 1024.00         | 56.0       | 2016-08-26 21:35:18 |
| 16.9266666  | 1024.00         | 54.6       | 2016-08-26 21:36:16 |
| 17.2916666  | 1025.00         | 56.0       | 2016-08-26 21:37:29 |
```

Fig. 4. Part of measurements in the database.

Other measurements did not require additional solutions because of their repeatability and reliability, as shown in Fig. 4. Also, the result for the temperature, pressure and humidity was the value obtained from a single measurement.

The main drawback of the presented system, compared to similar solutions in this field is the weather-sensitive hardware architecture (shield, ventilation). It may cause problems with testing under low temperatures, high winds or high humidity conditions such as steam condensation or freezing of components (short-circuits, problems with stability). In the next planned version of the system, amendments will be made to improve these aspects of the system.

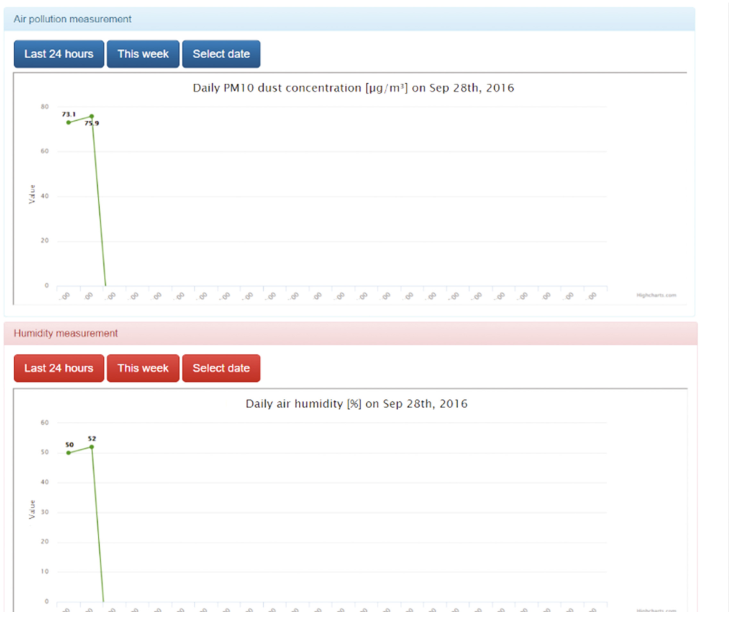

Air pollution measurement
Last 24 hours
This week
Select date
Daily PM10 dust concentration [µg/m³] on Sep 28th, 2016
Value
73.1
75.9
Humidity measurement
Last 24 hours
This week
Select date
Daily air humidity [%] on Sep 28th, 2016
Value
50
52

Fig. 5. Values of pollution, humidity monitored by the system.

4 Calibration and Performance Tests

The development of the prototype version of the SmogoMetr system allowed to carry out a series of tests of main functionalities of the system.

4.1 Display Measurements and Their Location

The first and one of most crucial tests performed right after the completion of system prototype, was to check that the system performs measurements of all parameters, writes them to the server and displays them correctly on the graphical user interface, taking into account their GPS coordinates. The presentation of these functionalities is presented in Figs. 5 and 6.

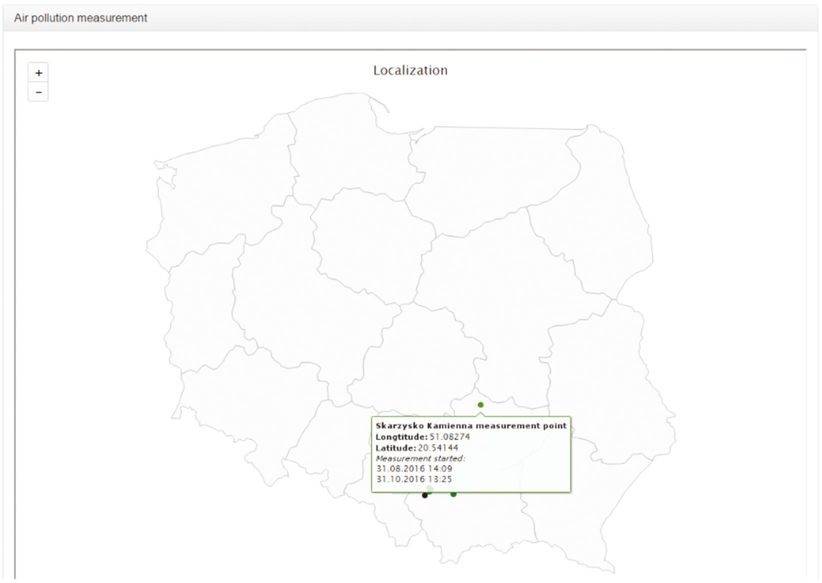


Fig. 6. Localization of measurement points.

In Fig. 5 values of air pollution and humidity parameters are presented. Values of the following parameters, which are not shown in the drawing, are placed in the interface below humidity graph.

Figure 6 shows the map of Poland, with the coordinates of the three measurement points in which the tests were performed. When user clicks on a particular measurement point, the interface returns measurement data from the selected location.

4.2 Sensor's Measurement Precision Testing and Calibration

One of the most important experiments was to verify that the values indicated by the sensors were consistent with actual values. The basic calibration of the measurements in the vicinity of Krakow Balice Airport, where a meteorological station performs measurements of parameters such as temperature, pressure and humidity. The second place where the accuracy of the sensors was tested, was area of Kurdwanow air-pollution measurement station managed by the Provincial Inspector of Environmental Protection in Cracow, where the compliance of the air-pollutant measurement with the measurement station was checked. The test of the UV light sensor was based on the estimation of the results on the basis of the values read on the Internet page of the Institute of Meteorology and Water Management for Katowice and Zakopane, because of the shortest distance from Krakow

Tests for the first three sensors (temperature, pressure, humidity) were successful. The digital barometer required improvement, due to a value that was about 25 hPa lower than referenced device. Other sensors worked properly, resulting a deviation below 1%.

The air-pollution sensor test consisted of ten measurements at two minute intervals between each measurement. Then the mean and standard deviation were calculated. The next step was to calculate the value of the divisor factor by dividing the obtained from the average by the expected value taken from the WEB portal WIOŚ (referenced device). The resulting factor was encoded in the prototype script as a divisor of the measured value to calibrate the sensor. The last step was to re-execute ten measurements to verify the correctness of the solution used.

The results are presented in Table 1. The tests were successful, the mean value after the divisor factor was considerably close to the expected value while decreasing the standard deviation.

Table 1. Dust sensor calibration, expected value $22\frac{\mu g}{m^3}$.

Original measurements		Calibrated measurements	
ID	Value	ID	Value
1	*215,7*	1	21
2	*204,6*	2	23
3	*205,8*	3	22
4	*204,3*	4	23
5	*190,2*	5	22
6	*194,7*	6	24
7	*195,9*	7	23
8	*207,6*	8	23
9	*181,8*	9	24
10	*194,1*	10	22
Average	*199,47*	Average	22,7
Standard deviation	*7,029*	Standard deviation	0,9

4.3 Dust Sensor Re-Testing

The greatest importance during prototype testing was attached to the dust sensor. Therefore, to ensure reliable and repeatable measurements, after the sensor calibration phase, it was decided to conduct tests at two subsequent measurement stations in Krakow. For this purpose, the measurement points of the Inspectorate at ul. Starowiślna and Nowa Huta, due to easy access to the stations. PM-10 Air-pollution values returned from prototype compared with the values read from the WIOŚ website in Cracow, which presented values from referenced devices. Tests were performed directly at the measuring stations, several meters away from the reference measuring station.

The results presented in the table (Table 2) confirm the reproducibility and reliability of SmogoMetr measurements..

Table 2. Dust sensor tests, after calibration.

NowaHuta measurement station, expected value $64 \frac{\mu g}{m^3}$		Starowiślna measurement station, expected value $36 \frac{\mu g}{m^3}$	
ID	Value	ID	Value
1	*62,5*	1	37,0
2	*54,2*	2	35,7
3	*68,1*	3	37,4
4	*67,6*	4	35,9
5	*63,8*	5	35,7
6	*66,0*	6	33,0
7	*72,9*	7	36,3
8	*60,2*	8	38,1
9	*62,4*	9	33,5
10	*63,9*	10	37,4
Average	*64,16*	Average	36,0
Standard deviation	*4,8*	Standard deviation	1,58

5 Summary

This paper presents the design, implementation and initial results of a prototype system that enables accurate measurements of air pollution, pressure, humidity, temperature and UV intensity. The system was calibrated based on existing, accurate, high-budget installations for monitoring these parameters, and in the testing phase showed high convergence of results with reference devices. The system can be used as an alternative to expensive, commercial solutions used by large organizations such as the Provincial Environmental Protection Inspectorate. The system has an additional advantage over the aforementioned devices, due to wireless communication, internal power supply source and compact size. This allows to carry out measurements at any location, while adding additional measurement values – GPS coordinates of the measurement system.

The presented prototype can be a basis for further development towards the creation of a sensor network based on elements modeled on the prototype. This would enable to perform tests at different locations at the same time, and to examine environmental variable influences such as altitude, wind speed and other environmental parameters that cannot be effectively tested using a single device. The prototype can also be considered as a low-cost environment monitoring station for households, which may be an attractive Internet of Thing module extension for Personal Area Networks solutions for monitoring of air pollution in urban environments.

Acknowledgements. The work has been supported by the Polish Ministry of Science and Higher Education – AGH University of Science and Technology Funds No. 11.11.110.300

References

1. Khedo, K.K., Perseedoss, R., Mungur, A.: A wireless sensor network air pollution monitoring system. arXiv preprint arXiv:1005.1737 (2010)
2. Al-Ali, A.R., Zualkernan, I., Aloul, F.: A mobile GPRS-sensors array for air pollution monitoring. IEEE Sens. J. **10**(10), 1666–1671 (2010)
3. Tsujita, W., Yoshino, A., Ishida, H., Moriizumi, T.: Gas sensor network for air-pollution monitoring. Sens. Actuators, B **110**(2), 304–311 (2005)
4. Kularatna, N., Sudantha, B.H.: An environmental air pollution monitoring system based on the IEEE 1451 standard for low cost requirements. IEEE Sens. J. **8**(4), 415–422 (2008)
5. Hartung, C., Han, R., Seielstad, C., Holbrook, S.: FireWxNet: a multi-tiered portable wireless system for monitoring weather conditions in wildland fire environments. In: Proceedings of the 4th International Conference on Mobile Systems, Applications and Services, pp. 28–41. ACM (2006)
6. Mainwaring, A., Culler, D., Polastre, J., Szewczyk, R., Anderson, J.: Wireless sensor networks for habitat monitoring. In: Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications, pp. 88–97. ACM (2002)
7. Kluska-Nawarecka, S., Regulski, K., Krzyżak, M., Leśniak, G., Gurda, M.: System of semantic integration of non-structuralized documents in natural language in the domain of metallurgy. Arch. Metall. Mater. **58**(3), 927–930 (2013)
8. Nychka, D., Saltzman, N.: Design of air-quality monitoring networks. In: Case Studies in Environmental Statistics, pp. 51–76. Springer US (1998)
9. Zampolli, S., Elmi, I., Ahmed, F., Passini, M., Cardinali, G.C., Nicoletti, S., Dori, L.: An electronic nose based on solid state sensor arrays for low-cost indoor air quality monitoring applications. Sens. Actuators, B **101**(1), 39–46 (2004)
10. Devarakonda, S., Sevusu, P., Liu, H., Liu, R., Iftode, L., Nath, B.: Real-time air quality monitoring through mobile sensing in metropolitan areas. In: Proceedings of the 2nd ACM SIGKDD International Workshop on Urban Computing, p. 15. ACM (2013)
11. Harkat, M.F., Mourot, G., Ragot, J.: An improved PCA scheme for sensor FDI: application to an air quality monitoring network. J. Process Control **16**(6), 625–634 (2006)
12. Malec, A., Borowski, G.: Zagrożenia pyłowe oraz monitoring powietrza atmosferycznego. Ecol. Eng. **50**, 161–170 (2016)
13. Fang, S., Da Xu, L., Zhu, Y., Ahati, J., Pei, H., Yan, J., Liu, Z.: An integrated system for regional environmental monitoring and management based on internet of things. IEEE Trans. Industr. Inf. **10**(2), 1596–1605 (2014)
14. Mead, M.I., Popoola, O.A.M., Stewart, G.B., Landshoff, P., Calleja, M., Hayes, M., Lewis, A.: The use of electrochemical sensors for monitoring urban air quality in low-cost, high-density networks. Atmos. Environ. **70**, 186–203 (2013)

Swarm Based System for Management of Containerized Microservices in a Cloud Consisting of Heterogeneous Servers

Waldemar Karwowski(✉), Marian Rusek, Grzegorz Dwornicki, and Arkadiusz Orłowski

Faculty of Applied Informatics and Mathematics, Warsaw University of Life Sciences, Nowoursynowska 166, 02–787 Warsaw, Poland
waldemar_karwowski@sggw.pl

Abstract. Centralized and hybrid container orchestration systems are a bottleneck for the scalability of cloud applications. Due to data replication costs the cluster can consist only of a handful of servers. A decentralized peer-to-peer systems are needed. We propose such a system whose architecture is the same as the microservice architecture of the cloud application it manages. It can potentially offer performance improvements with respect to the existing centralized container orchestration systems.

Keywords: Cloud computing · Swarm intelligence · Virtualization containers

1 Introduction

Traditional enterprise applications were monolithic and typically ran on a single computer. Over time, the enterprise systems became bigger and more sophisticated. The problems associated with implementation of large systems led to the componentization, applications began to be built as a collection of related modules. However, the modules were closely coupled. The development of networks has enabled the service-oriented architecture which generally is based on cooperation of loosely coupled modules. Despite this, the relationships between modules were still too strong. Too strong coupling led to cascading failures: one failing service could take down the entire system. Nowadays software architecture based on microservices [1] became more flexible solution. This architecture advocates creating a system from a collection of small, isolated services, resilient to failure and own their data. Each microservice is independently scalable. System based on microservices is far more flexible than a typical monolithic system. One of the important issues is the decomposition of the system into small isolated subsystems communicating over well-defined asynchronous protocols and decoupled in time and space. Another important task is the creation of infrastructure for monitoring and management microservces based application. Examples of such microservices orchestration tools are Google Kubernetes [2], Apache Mesos [3], and Docker Swarm [4].

The actual realization of a distributed orchestration system in the cloud requires that we instantiate and place software components responsible for microserevices monitoring and management on real machines. In traditional centralized architectures a single server or a group of servers implements most of the software components. In decentralized peer-to-peer architectures all nodes more or less play equal roles. Many real-word distributed systems are often organized in a hybrid fashion combining elements form both centralized and decentralized architectures.

Docker Swarm and Kubernetes are built using a centralized architecture. When the Docker Engine runs in swarm mode, manager nodes implement the Raft Consensus Algorithm to manage the global cluster state. The reason why Docker Swarm is using a consensus algorithm is to make sure that all the manager nodes that are in charge of managing and scheduling tasks in the cluster, are storing the same consistent state. Having the same consistent state across the cluster means that in case of a failure, any Manager node can pick up the tasks and restore the services to a stable state. For example, if the Leader Manager which is responsible for scheduling tasks in the cluster dies unexpectedly, any other Manager can pick up the task of scheduling and re-balance tasks to match the desired state. For performance reasons Docker recommends maximum 7 manager nodes, 3 from which may fail. Mesos employs hybrid architecture. There is a central manager as well as a distributed system of frameworks responsible for application scheduling. Each application needs a separate framework.

It is known that cloud based decentralized architectural models offer better performance than traditional Client-Server, Peer-to-Peer, and Hybrid architectures (see e.g. [5]). In connection to this, in this paper we propose a completely decentralized microservice management system based on swarm algorithm and analyze its performance on non-homonegeous hosts in different scenarios. Its system architecture is the same as the architecture of the underlying application. In practical realization our system can act as a second-level scheduler for Mesos. In this case no separate framework for each application would be needed because the application will automatically act as a framework.

This paper is organized as follows: in Sect. 2 the concept of a live migration of virtualization containers is presented. In Sect. 3 a swarm-like algorithm of container migration in the cloud is introduced. In Sect. 4 some preliminary experimental results for a simple cloud consisting of three hosts are given. We finish with summary and brief remarks in Sect. 5.

2 Live Migration of Containers

The container is a group of processes isolated from the rest of the system. The use of containers is one of the methods of microservices implementation [6]. Interest in the containers is associated with their performance. Public cloud virtual machines have to boot a full operating system and for every time it takes up to several minutes. In contrast to this the startup time for a container is around a second. Containers are ideal for the implementation of the independent parts of the system. This technology is not new but before Docker there was no common interface for container management and there was no standard format of how container is described in the system. However, using Docker we cannot provide live migration. Docker was created for different purpose, an automated application deployment, but there is commonly-used, container-based virtualization software, Parallels Containers, which supports live migration [7]. The results discussed in this paper were obtained using OpenVZ, an open source version of Parallels Containers.

Container live migration allows a user to start a microservice on any node of the cloud after which it can transparently move to other nodes to make efficient use of resources. The basic idea is that overall system performance can be improved if microservices are moved from heavily loaded to lightly loaded machines. However instead of offloading machines we can imagine that code is moved to make sure that a machine is sufficiently loaded. For example migrating complete virtual machines to lightly loaded machines in order to minimize the total number of nodes being used is a common practice in optimizing energy usage in data centers [8]. Another advantage of live migration is the ability to put the code processing the data close to where the data reside. This allows minimizing communication what is an important issue in todays distributed cloud environments [9]. In this paper we try to model all these scenarios by assigning different capacities to the hosts.

3 Swarm Algorithm for Non-homogeneous Hosts

The question of the optimal distribution of the containers on the hosts can have multiple variants. One of them is the uniform distribution of the containers on the hosts. This scenario is the following: we have S hosts (servers) and P containers (processes). At the start containers are arranged in a way that there is not the same number of them on every host. Uniform distribution means that the targeted number of containers on every host is equal $n = P/S$, and the goal of the algorithm is to achieve a state of balance. A decentralized algorithm for controlling the uniform distribution of tasks between nodes of a simple computational cloud was presented in [10]. In that approach, mobile virtualization containers use only local interactions and no communication to give the desired global behavior of the cloud providing dynamic load balancing between the servers. Each container only knows how many hosts S we have and how many other containers P are there, and can calculate the number N of neighboring containers located on the same host. If $N > n$ the container tries to migrate on another, randomly chosen, host. The probability of this migration is given by the equation $p = (N-n)/N$. The modified algorithm was presented in [11], and it offers potential improvement of performance over centralized orchestration systems. In this approach each host is described by a pheromone p which can be either repulsive $0 < p < 1$ or attractive $p < 0$. Attractiveness or repulsiveness is associated with the number of containers already on the host. In practice pheromone is equal to migration probability of a container. This probability can be modified $p = (N-n-Q)/(N-Q)$, where Q is the number of containers queued for migration in the kernel queue.

In above mentioned papers, it was assumed that the containers are similar and have almost the same memory requirements. An analogous assumption was made for the hosts: it was adopted that performance of each host is similar and we can treat hosts as homogeneous. However, in cloud environments, hosts can differ regarding CPU, number of cores and memory. Hence, optimization towards an equal distribution of tasks across available hosts does not seem to be optimal in each case. We will consider the situation when host performance is varied among hosts. Regardless of the actual performance of hosts, we may want to limit the burden of the selected host, for example, in order to exclude it from the operation of maintenance.

In this section we describe a swarm-like algorithm for containers migration in a cloud consisting of heterogeneous hosts. We assume that each container knows how many hosts we have and how many other containers are there, can calculate the number of neighboring containers located on the same host, and reads the available performance of this host. In addition, container can read the available performance of the host to which it intends to migrate and can count attractiveness or repulsiveness of such host. The container can update these values dynamically at runtime by probing other containers and hosts using ICMP echo/reply protocol. This protocol is implemented by ping. The available performance of the i-th host is denoted as h_i (note that h_i are different for different hosts). We can count the whole performance of all hosts as $\sum h_i$. Let us denote by c_i the number of containers on i-th host at the beginning, hence the whole number of containers P is equal to $\sum c_i$. Consequently the target number of containers on i-th host is

$$e_i = \frac{\mathrm{P}^* \, h_i}{\sum h_i}. \tag{1}$$

Let us denote by N_i the number of containers on i-th host during the preparation to migration. Hence the probability of migration is given by the equation

$$p = \frac{N_i - e_i}{\sum N_i}. \tag{2}$$

Taking into account the containers ready to migrate this probability can be modified as

$$p = \frac{N_i - Q_i - e_i}{\sum (N_i - Q_i)} \tag{3}$$

where Q_i is the number of containers queued for migration in the kernel queue from i-th host. The complete algorithm executed by a dedicated process running inside the container is the following [10, 11]:

```
 1: loop
 2:   Count the pheromone value p of the host.
 3:   Generate a random number 0 < r < 1.
 4:   if r < p
 5:     repeat
 6:       Randomly choose a host.
 7:       Get the pheromone value of the chosen host.
 8:     until chosen host has an attractive pheromone
 9:     Ask the host to migrate to chosen host.
10:     Wait until migration to another host is complete.
11:   end if
12: end loop
```

To ensure the autonomous operation of containers we have added a special system function to the hosts' operating system kernel. Thanks to this containers can initiate their migration. In the algorithm presented above we assume, that this call (performed in step 9) is a no blocking.

4 Experimental Results

The experiments were performed on 3 hosts equipped with Intel Dual Core E-5500 CPU, with potentially 2 GB of RAM, each connected by a dedicated Fast Ethernet network. All hosts were running Debian GNU/Linux operating system with OpenVZ software installed. The Linux kernel was modified by adding some new system functions including one mentioned in Sect. 3. Available host memory was chosen as a measure of host performance. The experiment was divided into three scenarios.

The first scenario corresponds to the situation when it is necessary to disable a single host, for example, for maintenance or repair. At the initial time 24 = 3*8 identical containers were launched on every host, 8 on each. Each container had a size in memory of about 55 MB. From measurements we know that its migration to another host takes about T = 12 s. Starting all the containers took about 1 min i.e. 5 T. In this scenario the performance values of all hosts were set to 0, 2, and 2 GB respectively. This means that the first host should be freed. After starting on all containers, the Python script read start values and counted target number of containers, in this scenario they were respectively: 0, 12, and 12 containers. After that script entered a loop (algorithm steps 1-12) in which it counted the probability of migration (step 2) and decided with probability p whether to migrate to another host. Next in the loop, container counted the pheromone value (step 7) of host for which it would like to migrate, and if host was attractive, container was attached to the migration queue. On the host a monitor program was running and periodically (period 5 s) checked the number of containers N in the filesystem and the number of containers queued for migration Q in the kernel queue (access to this data from a user process was possible by a custom system function added to the kernel). These data were available for containers. More than two hundred tests for the first scenario were performed and several typical results were identified. Generally they can be grouped and classified as regular and irregular.

In Fig. 1 we have typical regular case. It is seen from inspection of this plot that the containers can arrive to the destination hosts in parallel thus network bandwidth was apparently not a problem during this experiment. We notice that real migration process started around t $\simeq$ 5 T and ended around t $\simeq$ 16 T. A few time discrepancies are visible: host 1 (red) reached zero later ($\simeq$ 16.5 T) than host 2 (green), which reached target value $\simeq$ 15.5 T. It can be explained by some delay in checking.

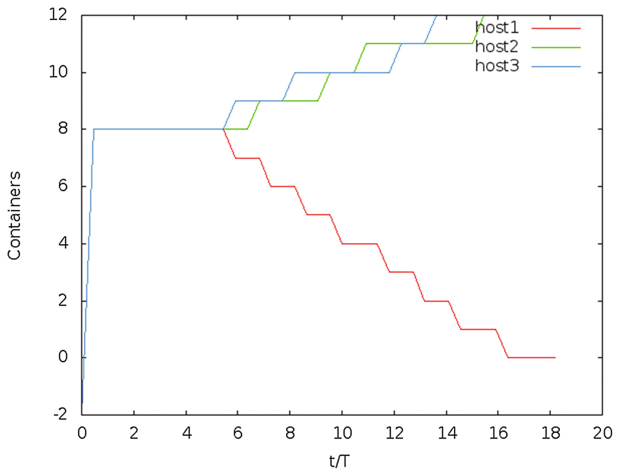


Fig. 1. Number of containers on each host versus time. The first scenario - regular case.

In Fig. 2 we have the example of irregular case. We notice, as before, that real migration process started around t $\simeq$ 5 T and ended around t $\simeq$ 31 T. Number of containers at host 2 (green) exceeded the target number between t $\simeq$ 12 T and t $\simeq$ 25 T. Later on number of containers at host 3 exceeded expected number between t $\simeq$ 26 T and t $\simeq$ 31 T. Finally system was stabilized. Two exceeds occurred because the migration process was inherently a probabilistic one. At the same time more than expected containers decided to migrate into host 2. Because the number of containers on host 2 reached too high value there had been a migration in the opposite direction into host 3. Probabilistic nature of algorithm caused that too many containers migrated into host 3, but finally target values were reached. In this scenario 27% of cases were regular, 73% irregular. Stabilization time for all regular cases ranged between 16 T and 18 T. Stabilization time for irregular cases was longer and ranged between 17 T and 40 T. However, three quarters of irregular cases were contained within the range [20 T, 28 T].

The second scenario corresponded to the situation when we have non-homogeneous hosts. Such situation may be caused by the need to free some resources on particular hosts. At the initial time 24 identical containers were launched, 8 on every host exactly like in the first scenario. Each container had a size in memory of about 55 MB. This time the performance values of all hosts were set to values 0.3, 1.2, and 2.0 GB respectively.

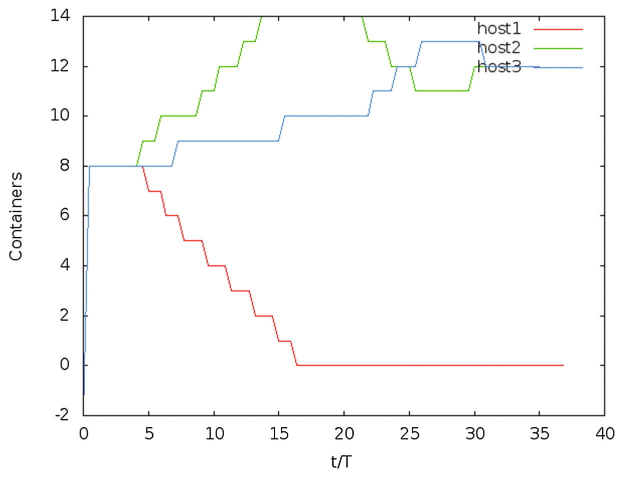


Fig. 2. Number of containers on each host versus time. The first scenario - irregular case.

In this scenario target numbers of containers were: 2, 8, and 14. More than one hundred tests for the second scenario were performed. Results in this scenario are very regular.

Typical result is presented in Fig. 3. The number of containers on the host 2 remained unchanged (we have to notice that pheromone of host 2 was 0) and containers from host 1 did not even attempt to migrate into host 2. Migration proceeded only from the host 1 to host 3. All cases were regular and more than 81% of them were contained within the range [12 T, 14 T]. Only a few had a longer stabilization time, the latest stabilization has occurred for 20T.

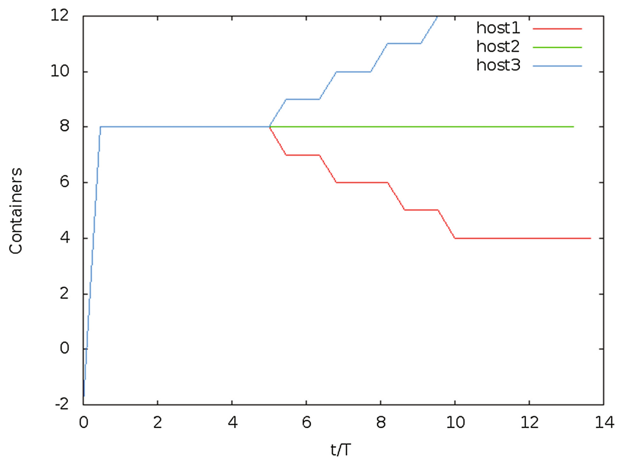


Fig. 3. Number of containers on each host versus time. The second scenario.

Third scenario was a bit different than two previous ones. At the initial time 24 identical containers were launched, 6 on the host 1, 12 on the host 2, and 6 on the host

3. Each container had a size of about 55 MB. The performance values of all hosts were set to values 2, 1, and 2 GB respectively. In this scenario target numbers of containers were respectively: 10, 4, and 10. More than eight hundred tests for the third scenario were performed. Three groups of typical results were identified. The first group can be classified as regular cases. Typical regular case is presented in Fig. 4. The migration process occurred only from host 2. It started around t $\simeq$ 5 T and ended around t $\simeq$ 15 T.

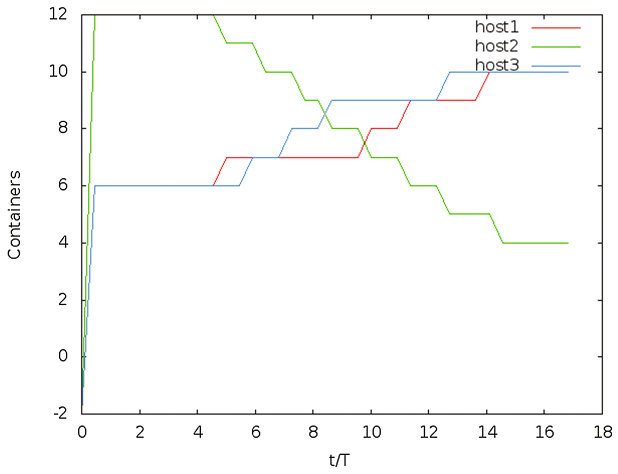


Fig. 4. Number of containers on each host versus time. The third scenario – regular case.

The second group of tests can be classified as slightly irregular cases. Typical slightly irregular case is presented in Fig. 5. The migration process occurred only from host 2

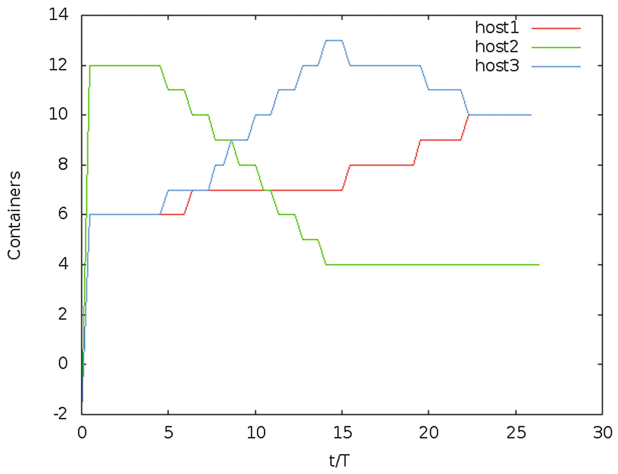


Fig. 5. Number of containers on each host versus time. The third scenario – slightly irregular case.

as before. It started, as previously, around t ≃ 5 T but generally ended later than t ≃ 20 T (22 T in Fig. 5). The temporary exceeding of the target number of containers occurred only on one host (host 1 or host 3; in Fig. 5 host 3) and was connected with probabilistic character of migration.

The third group can be classified as strongly irregular cases. Typical situation from this group is presented in Fig. 6. The migration process, as before, occurred only from host 2. It started around t ≃ 5 T but ended much later than in the second group, typically about t ≃ 30 T (32 T in Fig. 6). This case is strongly irregular because overrun took place both on host 1 and host 3. As we see in Fig. 6, firstly number of containers on host 1 exceeded target number, after that (around t ≃ 20 T) number of containers decreased on host 1 and increased on host 3. Finally situation was stabilized about t ≃ 32 T. The apparent stabilization (around t ≃ 22 T), results from certain delays in checking.

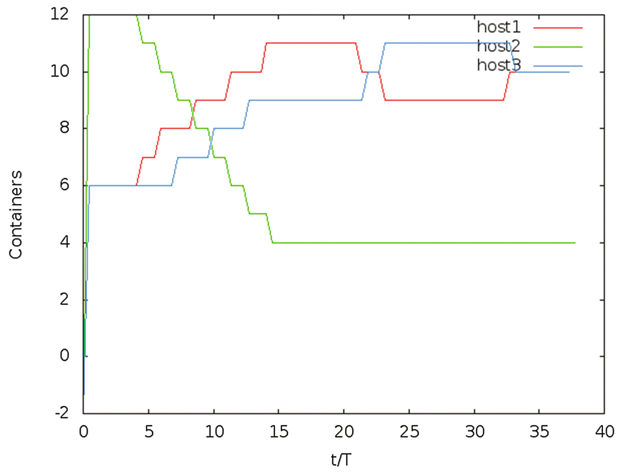


Fig. 6. Number of containers on each host versus time. The third scenario – strong irregurality.

In third scenario 28% of tests were regular, 64% slightly irregular and 8% strongly irregular. Stabilization time for all regular cases ranged between 15 T and 17 T. Stabilization time for slightly irregular cases was longer and ranged between 18 T and 38 T. However, nine-tenths of slightly irregular cases were contained within the range [18 T, 27 T]. Stabilization time for strongly irregular cases was much longer and ranged between 25 T and 50 T.

5 Conclusions

The aim of this work was to examine self-organized migration of containers in a heterogeneous environment. The swarm-like algorithm prepared for this task was tested on a small "cloud" consisting of three nodes. Three scenarios were examined. In the first scenario initially all hosts were equally loaded on startup and one host had to be turned off. In the second scenario, at the beginning all hosts were equally loaded on startup but ultimately, the load on the hosts has to be diversified. In the third scenario both at the beginning and at the end the load on the hosts was different. This time it was necessary to migrate from the host of the most heavily loaded to hosts initially less loaded. The performance of the algorithm was satisfactory in all three cases. The best performance of the algorithm was in the second scenario. To further improve the algorithm, in a future work we plan to choose the host to migrate to when leaving the migration queue and not when entering it as in this paper. Such an improved algorithm will be not only tested experimentally for small number of hosts but also simulated using a cellular automaton-like model. Such simulation would allow us to study the scaling of the algorithm with number of hosts and compare it to the behavior of typical clouds consisting of thousands of hosts.

References

1. Thönes, J.: Microservices. IEEE Softw. **32**(1), 116 (2015)
2. Brewer, E.A.: Kubernetes and the path to cloud native. In: Proceedings of the Sixth ACM Symposium on Cloud Computing, pp. 167. ACM (2015)
3. Hindman, B., Konwinski, A., Zaharia, M., Ghodsi, A., Joseph, A.D., Katz, R.H., Shenker, S., Stoica, I.: Mesos: a platform for fine-grained resource sharing in the data center. In: Proceedings of NSDI 2011: USENIX Symposium on Networked Systems Design and Implementation, pp. 295–308. USENIX (2011)
4. Stubbs, J., Moreira, W., Dooley, R.: Distributed systems of microservices using Docker and Serfnode. In: 7th International Workshop on Science Gateways (IWSG), pp. 34–39. IEEE (2015)
5. Gital, A.Y.U., Ismail, A.S., Chiroma, H., Abubakar, A., Abdulhamid, B.M.A., Maitama, I.Z., Zeki, A.: Performance analysis of cloud-based CVE communication architecture in comparison with the traditional Client Server, P2P and Hybrid models. In: Information and Communication Technology for The Muslim World (ICT4 M 2014), pp. 1–6. IEEE (2014)
6. Pahl, C.: Containerization and the PaaS Cloud. IEEE Cloud Comput. **2**(3), 24–31 (2015)
7. Mirkin, A., Kuznetsov, A., Kolyshkin, K.: Containers checkpointing and live migration. In: Proceedings of Linux Symposium, vol. 2, pp. 85–90 (2008)
8. Zhao, M., Figueiredo, R.J.: Experimental study of virtual machine migration in support of reservation of cluster resources. In: Proceedings of the 2nd International Workshop on Virtualization Technology in Distributed Computing, p. 5. ACM (2007)
9. Kratzke, N.: About microservices, containers and their underestimated impact on network performance. In: CLOUD COMPUTING 2015: The Sixth International Conference on Cloud Computing, GRIDs, and Virtualization, pp. 165–169 (2015)
10. Rusek, M., Dwornicki, G., Orłowski, A.: Swarm of mobile virtualization containers. In: Świątek, J., Borzemski, L., Grzech, A., Wilimowska, Z. (eds) Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part III. Advances in Intelligent Systems and Computing, vol. 431, pp. 75–85. Springer, Cham (2016)
11. Rusek, M., Dwornicki, G., Orłowski, A.: A decentralized system for load balancing of containerized microservices in the cloud. In: Świątek, J., Tomczak, J. (eds.) Advances in Systems Science, ICSS 2016. Advances in Intelligent Systems and Computing, vol. 539, pp. 142–152. Springer, Cham (2017)

Execution Management of Computational Services in Service-Oriented Systems

Paulina Kwaśnicka, Łukasz Falas, and Krzysztof Juszczyszyn(✉)

Wrocław University of Technology, Chair of Computer Science, Wyb. Wyspiańskiego 27, 50-357 Wrocław, Poland
{paulina.kwasnicka,lukasz.falas, krzysztof.juszczyszyn}@pwr.wroc.pl

Abstract. The aim of this work is to propose effective solution to eliminate the problem of excessive resource locking by idle computational services. In order to achieve this, a dedicated execution scheme and software tools were developed and tested in several scenarios. The results show that resource locking may be significantly decreased and the overall resource consumption in service system can be minimized as well. The approach is dedicated for computational services, which perform operations upon data delivery, returning the results after finishing their tasks.

Keywords: Service execution · Resource allocation · Service oriented architecture · Web services

1 Introduction and Related Work

Resource allocation problem in service systems in inherently connected with quality of services and fulfilling the client's demands. There exist many solutions, which evolved along with service-oriented architectures. For example, the Q-RAM model (QoS-based Resource Allocation Model) was developed in 1997 [1]. It assumes distribution of resources among applications with multi-criteria quality constraints (including security parameters), guaranteeing minimal resources needed to fulfill requirements to each one. Unfortunately, it does not address distributed environments and multi-machine resource locations.

Also, there exist solutions which rely on auction mechanisms [2]. In these approaches, participating subjects are divided into the sets of clients and service providers. All bids concerning resource allocation are directed to the central management module, called Auctioneer. It sorts queries in decreasing order, and offers in increasing order, and assigns tasks to the "highest bidder", just like it happens in stock market. The same mechanism, with some modifications, was applied to service cloud environments [3]. In this case, users are served by service provider (Cloud Service Provider – CSP). His task is to provide dynamic sharing of resources among cloud users, which may involve auction approach. In the system described in [3] a generalized second price algorithm was used, which means that offers are sorted in decreasing order, and bids are allocated to them starting from the highest, until offered resources are exhausted.

It should be noted, that auction mechanisms require additional functionalities which have to be provided to the clients. They must have interfaces for submitting bids and offers, there is also some processing overload, introduced by the auction system itself.

Some solutions address (and exploit) virtualization, starting from assumption, that a single virtual machine supports a service [4]. The resources are then assigned to the virtual machines, in order to maximize service efficiency. Such approaches typically use service types, which means that services are categorized and divided into groups, each relying on the extensive use of certain types of resources (computational, communication, memory, etc.). For example, in Windows Azure, types are used to match services with virtual machines, for which the amounts of available resources are known. The management module analyses (in time intervals) the requirements of the service queries, and compares them with the averaged pool of resources available at virtual machines. If there is a danger of not meeting the requirements of the service queries, it is possible to start new virtual machines. Services have also time constraints, and it was shown that they are fulfilled in 90% of the cases, with this approach [4].

Quasar [5] is a cluster management system, utilizing coordinated resource allocation and placement. It does not use resource reservation (reservation-centric approach), applying performance-centric techniques instead. Advanced classification schemes are used in order to evaluate the effectiveness of various coordination schemes, and information gathered is used to infer the optimal resource localization. In most cases it leads to almost complete utilization of available resources.

In [6] various models of directing service queries were evaluated. The distribution of resource consumption and probabilities of service availability was checked for various scenarios involving single and multiple servers with virtual machines and diverse pools of resources. In this approach idle service queries (waiting for resource allocation) were placed in queues.

Service-oriented QoS framework ROAR (Resource Optimization, Allocation and Recommendation System) concentrates on service applications' effectiveness [7]. The quality constraints are user-defined, and a special language (GROWL - Generic Resource optimization for Web applications Language) was designed for it. Its statements are parsed by the service management systems and after that, the Docker containers are set up for the service applications. Containers are also tested for their performance, and test results are used to assign services to containers.

At the moment, cloud platforms gain popularity in a big way, many of them advertising themselves as "capable of delivering any amount of resources, fulfilling client's demands". For example, Microsoft Azure offers "automated scaling" of resources [8, 9]. It allows client to choose from the predetermined set of types of virtual machines, and to define the timeframes, the client wishes to use them. There are also mechanisms for dynamic scaling of resources, according to the their actual consumption by the client's services. Similar approach is offered by Amazon Cloud Services [10], where the users can create groups of services and define the minimum and maximum numbers of their instances, along with conditional plans of rescaling these numbers. The same is possible in Google Cloud Platform [11].

Summarizing these efforts we can say, that there is a number of approaches, ready to apply, and there are also commercial products, utilizing them. However, there is one important problem outstanding – none of these solutions address the issue of resource locking, in cases where a service is active, and the resources are reserved for it. Computational services require significant amount of memory and computational power, but their actual resource consumption strictly depends on the number (and frequency) of queries. Thus, in many cases, we have running services, which waste the resources while being active (when their resources are allocated and locked, but there are none or small numbers of queries directed to them).

Taking this into account, this paper proposes a novel system, dedicated to the management of computational services with time constraints guarantees and dynamic resource allocation optimization.

The rest of the paper is organized as follows: Sect. 2 describes the proposed approach and the architecture of the system, Sect. 3 presents the formal definition of execution management problem, Sect. 4 introduces the proposed solutions for the defined problem, Sect. 5 depicts empirical research results and finally, Sect. 6 summarizes our research results and briefly presents plans for further research.

2 Service Management System

Basing on our experiences and the results from the works cited in the preceding section, we decided to use "serve on demand" approach (resources are allocated to services when they are needed). In most of the existing solutions, virtual machines (VM) are used to do so, but there is a factor of large delays, inevitable when the VM starts. Hence, in our case a Docker container technology was chosen. It was also decided to abandon the popular approach of keeping numerous VMs and starting a service on one of them. In our approach each service (with known amount of resources required to run it) is active, and there is a number of its instances (containerized). This also allows to create new instances of services, with predetermined amount of instances allocated. This prevents assignment of excessive resources to services (like in the case of running single-thread service on multicore VM).

Another feature of our approach is the priority of the effective query execution – time limits determine the configuration of the container which hosts the service. After service execution, the container is immediately shut down, freeing the resources allocated to it.

The architecture of our system requires at least two servers: Management Server, (MS) which is a center of the architecture, and one (or, in most cases, multiple) worker server on which the services are executed.

The outline of server configuration is presented in Fig. 1.

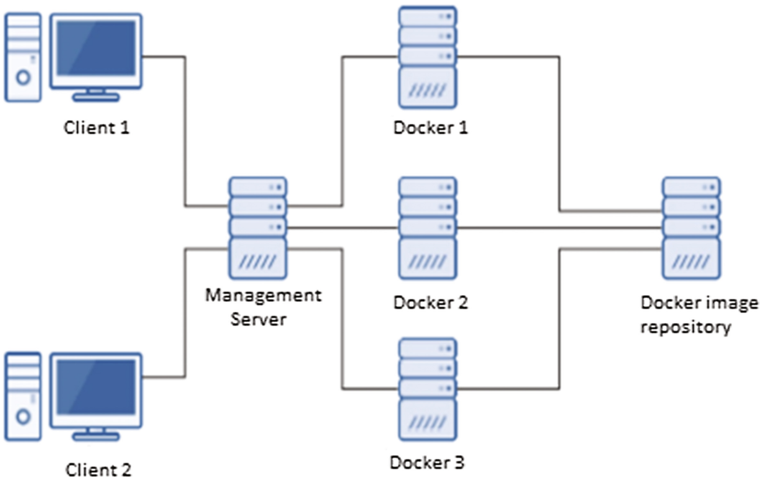
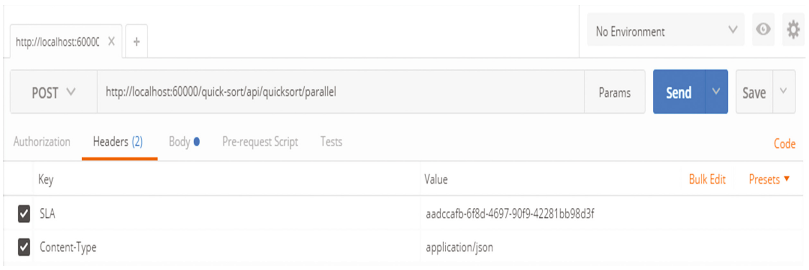


Fig. 1. Exemplary architecture, with managing and service-executing servers.

The clients generate computational service queries. They are unaware of the MS, and, from their point of view, the queries are sent directly to the services. The MS is transparent for the clients, hence they can use any standard client (e.g. Postman or one of programming libraries with http client) for sending their requests to web services available in the system. A sample request (in Postman) with dedicated http header identifying SLA connected to the request is presented in Fig. 2.



POST http://localhost:60000/quick-sort/api/quicksort/parallel

Key	Value
✓ SLA	aadccafb-6f8d-4697-90f9-42281bb98d3f
✓ Content-Type	application/json

Fig. 2. A query directed to the MS via the Postman.

The MS gathers all of the queries and checks their time constraints (deadlines). Then, it uses a number of strategies (strategies are defined in the next section) to chose a server on which the container for the instance of a service will be created. After that, the query is directed to the container hosting the service. The MS was implemented in C#, .Net Core version.

All servers run the Docker virtualization platform (which, in contrary to the VM, takes less time to be created, and consumes less resources). The MS creates the containers for the services, and forwards client's query to them. Upon completion of the query container is shut down, freeing all of the resources (operations are executed via Docker API, version 1.26, in Docker version 1.13.1).

Docker containers are instances of images of file systems and execution environment's settings [14]. The images are being queried from Docker Hub repository or private Docker repository.

The MS mode of operation is presented in Fig. 3. For the clarity of presentation it does not involve request execution retries during error occurrence (for example, if the container cannot be created).

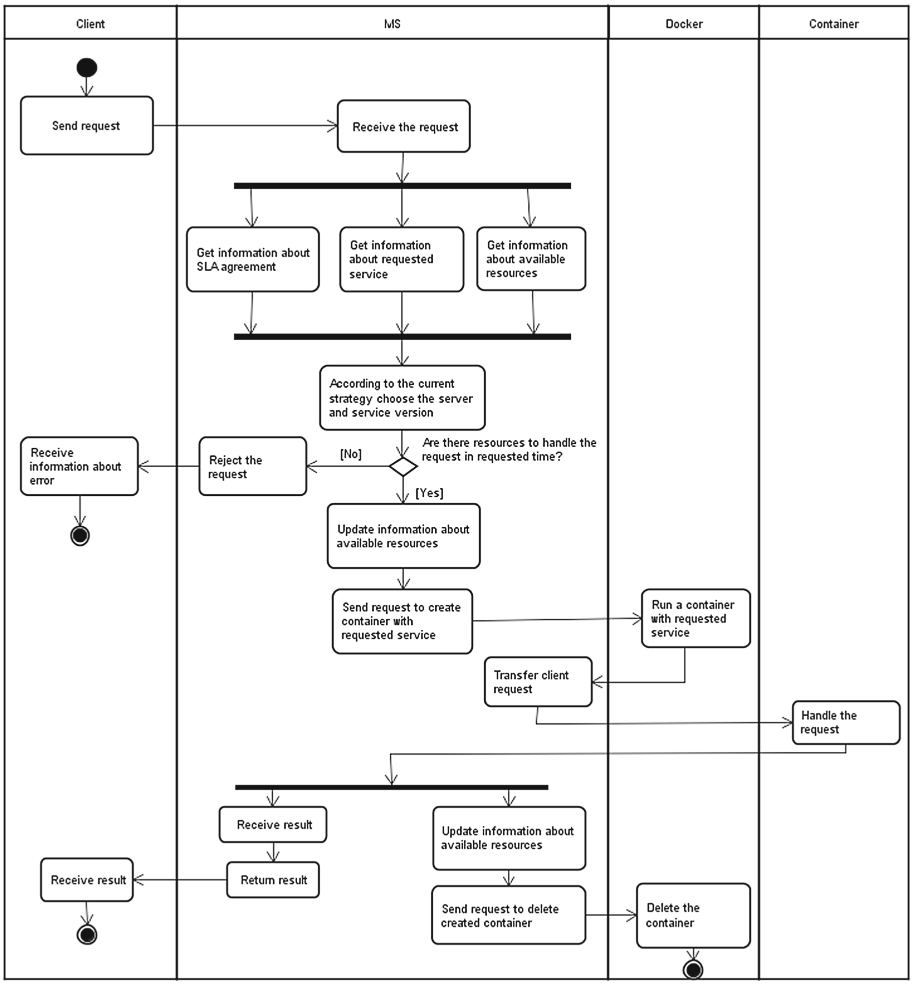

Client
MS
Docker
Container
Send request
Receive the request
Get information about SLA agreement
Get information about requested service
Get information about available resources
According to the current strategy choose the server and service version
Are there resources to handle the request in requested time?
[No]
[Yes]
Receive information about error
Reject the request
Update information about available resources
Send request to create container with requested service
Run a container with requested service
Transfer client request
Handle the request
Receive result
Update information about available resources
Receive result
Return result
Send request to delete created container
Delete the container

Fig. 3. The management server's workflow.

3 Problem Formulation

Let's define the set of servers:

$S = \{S_1, \ldots, S_n\}$, where n is the total number of servers available, and

$Z_n = \{Z_1, \ldots, Z_n\}$, is the number of available resources on the server (processors, cores, memory, etc.). We have also the set of services:

$U = \{U_{11}, \ldots, U_{1j}, \ldots, U_{i1}), \ldots, U_{ij}\}$, where i is the total number of services, and j is the number of versions of the i-th service.

The queries arrive at the time t. Each of them has a size of R expressed in bytes, unique ID i, and the SLA definition, containing the deadline T. There are two conditions for the query execution: its time should not exceed the deadline:

$$t_r + t_o \leq T \tag{1}$$

Where t_o is query processing time (which, in our case is constant, and equals to 1300 ms.), and t_r is an expected time of query execution, which is estimated by linear function:

$$f\left(U_{ij}\right) = \mathrm{a} + \mathrm{b} * \mathrm{R} = t_r \tag{2}$$

The parameters a and b are computed with linear regression method for each of the service's instance.

The second condition is the capability to run the service on the chosen server:

$$f_r\left(Z_n(t)\right) - f_r\left(U_{ij}\right) \geq 0 \tag{3}$$

$Z_n(t)$ stands for the server's available resources, and fr is a function which returns the resources needed to run the service. It is defined as follows:

$$f_r(Z(Uij)) = 0.5 * f_{nRAM}(Z(Uij)) + 0.5 * f_{nCPU}(Z(Uij)) \tag{4}$$

Where $Z(Uij)$ is the amount of resources needed to execute j-th versions of service i (defined in the same way as Z_n), f_{nRAM} returns a normalized (from 0 to 1) requirement for RAM, and f_{nCPU} returns a normalized number of processing cores.

Our aim is to find the most effective strategy for the MS, considering the key parameters: the number of queries rejected $\left(L_o\right)$, the number of queries not meeting the deadline $\left(L_s\right)$, average CPU usage $\left(\mathrm{Av}_{CPU}\right)$, and average RAM consumption RAM $\left(\mathrm{Av}_{RAM}\right)$. We have applied the following equation for the overall effectiveness:

$$f_e\left(L_o, L_s, \mathrm{Av}_{RAM}, \mathrm{Av}_{CPU}\right) = 0,4 * L_o + 0,3L_f + 0,3 * \frac{1}{n} \sum_{k=1}^{n} \left(\mathrm{Av}_{RAM}(k) + \mathrm{Av}_{CPU}(k)\right) \tag{5}$$

Which implies, the less effectiveness value means the strategy is better.

4 Algorithms Managing the Execution of Computational Services

In order to solve the problem of computational services execution management the original problem was decomposed into two sub-problems: (1) selection of a computational server on which service will be deployed and executed, (2) selection of one of service versions ensuring compliance with defined SLA (each version has different needs of resources).

For the selection of computational server two approaches were proposed:

- carousel – every request is launched on next server from server list.
- fill server – when resources on server run out, then change server to the next from the list

For the selection of service version the following approaches were and developed:

- fastest first – launch largest (in terms of required resources) possible service version, which ensures fastest request execution.
- minimize usage – launch smallest (requiring smallest amount of resources) service version which predicted execution time meets SLA.

Finally, through combination of these approaches, following strategies for managing the execution of computational services were developed:

- Strategy 1 – combination of carousel and fastest first approaches.
- Strategy 2 – combination of carousel and minimize usage approaches.
- Strategy 3 – combination of fill server and fastest first approaches.
- Strategy 4 – combination of fill server and minimize usage approaches.

5 Study of Developed System Efficiency

Empirical tests (simulations of clients' behavior) of proposed strategies have been performed to define the efficiency of proposed solution.

Hardware infrastructure consisted of eight worker virtual machines with Ubuntu 14.05.4 deployed on a blade server. To each one of them 8 GB of RAM and 4 processor core was assigned. Real time resource usage was measured with Docker software and an custom agent for resource usage information gathering which were installed on test machines. The prepared virtual machines with Docker software were acting as a small computing cluster in this research. The MS and test software were deployed on a dedicated management server with 8 GB of RAM and 8 processor cores.

The tests utilized 3 services with following algorithms: quicksort, matrices addition and knapsack problem solver. Each of the services utilized REST API and was deployed in three different versions presented in Table 1. For each service prepare 3 different service versions.

Table 1. Test service versions.

Algorithm	Quicksort			Add matrixes			Knapsack problem		
Version	S	M	L	S	M	L	S	M	L
RAM	1	2	4	1	2	2	1	2	2
CPU	1	2	4	1	2	3	1	2	4

Seven test were prepare to study system efficiency (Table 2). During each test 50 request was send to MS. Requests were randomly choose from set of 30 predefined requests. Each request in this set had a request url, data as JSON file and SLA agreement id.

Table 2. Test specification.

Test name	Test1	Test2	Test3	Test4	Test5	Test6	Test7
Number of clients	2	2	2	5	5	5	10
Minimum break between requests (ms)	1000	1000	1000	1000	1000	1000	1000
Maximum break between requests (ms)	10000	5000	1500	10000	5000	1500	1500

In order to simulate multiple clients' requests, in the dedicated test application each client was represented by a dedicated thread. Each of the threads after sending request to MS thread was put to sleep for randomly chosen time in range from minimum break between requests to maximum break between requests. Each of the tests has been conducted five times on two different clusters: (1) containing of 4

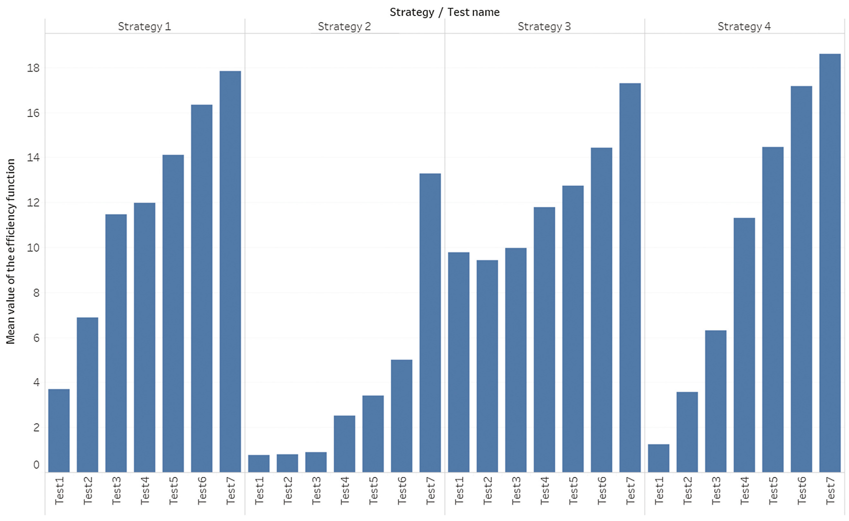


Fig. 4. Test results for 4 worker servers (mean value of efficiency function).

worker servers, (2) containing of 8 worker servers. The results of conducted experiments are presented below (Figs. 4 and 5).

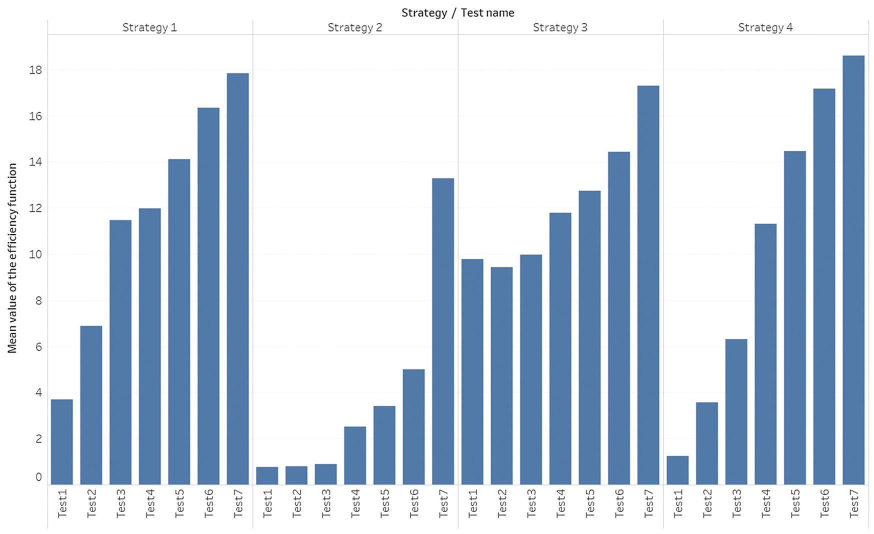


Fig. 5. Test results for 8 worker servers (mean value of efficiency function).

The comparison of proposed strategies (Figs. 4 and 5) shows that, as it was expected, the proposed strategies are able to achieve best values of the efficiency function for low rate request streams, while they are also able to prevent resource locking, which directly increases the average resource availability in the system. This characteristic was different only for the strategy 3 (combination of fill server and fastest first approaches), which had a stable value of efficiency function in all tests, except the last test with high rate request stream.

Also, the results clearly indicate that strategy 2 (combination of carousel and minimize usage approaches) was the most effective strategy, due to the fact that in all test cases this strategy achieved to minimal mean values of the efficiency function among all of the tested strategies. It was especially effective in case of low rate request streams and it provides resource locking prevention mechanisms.

6 Conclusions

The aim of our work was to propose an effective solution for resource locking by idle computational services. In order to achieve this, a novel and flexible mechanism of service execution management was designed and implemented. It uses (increasingly popular) container technology, which allows fast, on-demand, service configuration and start-up, along with the possibility of its fast removal from the system. Also, service versioning is used, and service instances may have different sets of resources assigned.

A number of service management strategies were designed and experimentally tested on real services, which allowed choosing the most effective ones.

Summing up, our solution minimizes the overall costs of service-oriented environments, and the most important factors contributing to this result are:

– elimination of idle services, running when they are not needed and locking resources,
– containers (assigned to user queries) have small costs of set-up and removal,
– multiple service instances increase resilience in the case of errors,
– effective service management strategies, and mediation of all queries by Manager Server (which hides dynamic addresses of the services).

Our approach is dedicated to service systems which do not deal with large number of user queries (coming every second), it rather addresses scenarios which involve significant possibility that the services will wait for queries, unnecessarily locking resources in the process.

The most important issues which will be worked on in the near future will be developing prediction methods to accurately estimate resource consumption of the services (which may depend on the size and the content of user queries), and methods for determining parameters and the number of the instances of any given service.

Our approach was tested on real-life examples of service repositories. Further research will be concentrated on adaptation of the developed methods to various service repositories and the development of dedicated algorithms for faster decision making for resource allocation. The next experiments will be run using bigger repositories and basing of the resources of recently set up Laboratory of Computing Clouds which is a part of national Polish PL-LAB2020 infrastructure.

References

1. Rajkumar, R., Lee, C., Lehoczky, J., Siewiorek, D.: A resource allocation model for QoS management. In: Real-Time Systems Symposium, pp. 298–307. IEEE Computer Society Washington, San Francisco (1997)
2. Izakian, H., Abraham, A., Ladani, B.: An auction method for resource allocation in computational grids. Future Generat. Comput. Syst. **26**, 228–235 (2010)
3. Lin, W., Lin, G., Wei, H.: Dynamic auction mechanism for cloud resource allocation. In: IEEE/ACM International Conference on Cluster 2010, pp. 591–592. Melbourne (2010)
4. Mao, M., Li, J., Humphery, M.: Cloud auto-scaling with deadline and budget constraints. In: IEEE/ACM International Conference on Grid Computing 2010, Brussels, pp. 41–48 (2010)
5. Delimitrou, C., Kozyrakis, C.: Quasar: resource-efficient and QoS-Aware cluster management. In: International Conference on Architectural Support for Programming Languages and Operating Systems 2014, Salt Lake City, pp. 127–144 (2014)
6. Vakilinia, S., Mustafa, A., Dongyu, Q.: Modeling of the resource allocation in cloud computing centers. Comput. Netw. **91**, 453–470 (2015)
7. Sun, Y., White, J., Li, B., Turner, A.: Automated QoS-oriented cloud resource optimization using containers. J. Syst. Softw. **116**, 146–161 (2016)
8. Azure auto scale. https://azure.microsoft.com/pl-pl/features/autoscale/. Last Accessed 29 Mar 2017

9. Azure auto scale – best practices. https://docs.microsoft.com/en-us/azure/architecture/best-practices/auto-scaling. Last Accessed 29 Mar 2017
10. Amazon auto scale. http://docs.aws.amazon.com/autoscaling/latest/userguide/WhatIsAutoScaling.html. Last Accessed 29 Mar 2017
11. Google Cloud auto scale. https://cloud.google.com/compute/docs/autoscaler/. Last Accessed 29 Mar 2017
12. Docker basic information. https://docs.docker.com/engine/getstarted/step_two/. Last Accessed 29 Mar 2017

Content Delivery Network and External Resources Detection for Selected Hosts

Vladyslav Deyneko(✉) and Ireneusz J. Jóźwiak

Wroclaw University of Science and Technology, Wroclaw, Poland
v_deyneko@outlook.com, ireneusz.jozwiak@pwr.edu.pl

Abstract. Many methods are used to reduce the time required to download web page completely. Most of them based on different techniques such as media elements and scripts compression, parallel resource download or caching. One of such techniques based on resource sharing and CDN usage. Resource sharing is an effective design for web pages with a considerable amount of media elements. In this paper, the author introduces his own software to determine, if selected web page uses CDN or external resources. In addition, questions about the correlation between web-page size, category and CDN usage are taken into account.

Keywords: Web performance · Content delivery network · External resources · Internet infrastructure

1 Introduction

One of the link quality estimation methods between user and selected web page is a page speed index. This concept has many influential factors and user interface performance is one of them. In case of desktop applications, interface performance is often defined by application architecture, when its organization and complexity are based on primary functionality. Desktop software usually works on the client side, what means, that software has to be installed on the machine before it can be used. Product installation is related to integration of its component into the system for requirements verification. Thus, interface performance depends on both program architecture itself and machine hardware configuration.

Web applications, in contrast to desktop software, require a stable Internet connection, because they are delivered through it. When a user enters a web page, he or she starts using all the needed functions without installing any software. The functionality of a web application is still based on its main intention and interface has to fulfill performance and flexibility requirements. On each stage of user interaction, graphical interface is a primary tool that firstly has to be delivered to the user device. The majority of services that are intended for measuring a website performance, have to download the entire page with external resources to analyze it and provide reasonable analytics. From this point, we can assume that the speed of user interface delivering (in other words, page load speed) in some sense is the most valuable characteristic of a web page [4]. That assumption means that the less time user will wait for the entire page to download the more comfortable and pleasant his experience will be.

For the webmasters, one of the most important web page parameters is page load speed. In web browsers, this metric can be calculated basing on the *DOMContentLoaded* event, which occurs when the entire HTML document is loaded and parsed without waiting for the external resources [1]: script files, CSS styles or media elements. Another very similar index is *Load* event [2] that fires when complete HTML document with external resources has been entirely downloaded. In theory, if *DOMContentLoaded* event usually fires just after 2–4 s after the user clicks on the link, the *Load* event can be delayed for a considerable amount of time because of waiting for external resources. If a web page does not use an *async* or *defer* attributes that define how the JavaScript file should be loaded [3], the browser will download all script files in a sequential order. In both cases, the *Load* event indicates a juncture when a web page is fully functional, and thus the most important for the user.

There are existing methods for reducing page load time. In this paper, we will discuss one of the oldest but effective technique to handle a huge amount of incoming requests but provide a very fast content delivery – CDN. Content Delivery Networks have received huge popularity among both clients and researchers in the last few years. Despite being an old technology (first networks of this type appeared in late 1990s [5]), CDN started to be a very effective and profitable solution with the growth of media streaming services such as Netflix, Spotify or YouTube. The use of "server farms" when many hosts were delivering content from one or few networks was not enough because of the main problem: many servers could not provide reasonable page load times if they were located in a great distance from user [6]. The architecture of content delivery networks requires being geographically dispersed to be as close to the end user as possible. From the web page perspective, the CDN technology is closely related to resource sharing which has to comply the specification of Content Security Policy, which implies how static resources can be delivered to the end user.

The rest of this paper is organized as follows. Section 2 gives a brief overview of the content delivery network architecture. Additionally, it describes base principles of cross-origin resource sharing. Section 3 describes the applied method that was used to obtain data for the analysis. The internals of implemented application that was used in present research is introduced in Sect. 4. Finally, Sect. 5 includes the conclusions from this paper. It outlines the future work as well.

2 Technology Stack Overview

2.1 External Resources and Content Security Policy

External resources usage is very common nowadays. With saturation of the Internet with media content, many web pages started to declare the majority of necessary assets as external elements. This technique helps to distribute required assets across multiple domains to improve both page loading time and reduce server load. As media elements, scripts, CSS styles and fonts are static resources, web site owners are able to separate static and dynamic content between multiple servers. This helps to improve server performance, which can be customized basing on a data character. Dynamic content is generated by the server in the moment of data request, usually by sending queries to the database retrieving necessary data. Static content is located on the server in the ready-made form, which means it is prepared to be used instantly. In this case, the server will send the identical response with the same content for all requests.

For the web page, static content is always a part of the page body. The HTML document, which describes the page structure, usually includes many required elements, which are loaded from external web pages. This mechanism is controlled by the Content Security Policy [7, 8] that defines the constraints of external resources fetching or execution. By using this technique, web page owners can limit and lock down their applications in a various ways, to prevent from content injection vulnerabilities to occur (for example, it can help to prevent Cross-Site-Scripting attacks). This policy can be applied to a wide variety of resources and is presented as a set of directives. Each of these directives has a number of associated algorithms and is responsible for controlling specific resource types and behaviors. By the time, when this article was written, available specification (Content Security Policy Level 3, 13 September 2016) includes following sets of directives:

- **Fetch directives** controls a whitelist of sources, from which an assets can be downloaded. For example, the HTML *Content-Security-Policy* header with a *default-src 'self'value* will tell the browser to fetch resources only from the same domain;
- **Document directives** regulates types of document properties and working environments, for which the policy applied. In this set, the `plugin-types` directive manages the set of plugins that can be embedded and used in the HTML document;
- **Navigation directives** are used to take control under the URLs that are used as targets for forms submission;
- **Reporting directives** defines target endpoints to which the reports will be sent when the error or policy violation occurs.

The available set of rules helps to organize flexible and effective control mechanism to manage resources sharing across the entire page. CSP is one of the primary methods to improve overall web page security.

2.2 Content Delivery Network

As was discussed in the previous paragraph, static and dynamic content create different server load types. Dynamic content creates high demand on CPU resources, I/O for the database and memory. Static content puts high demands on the network performance and on IO operations for non-cached resources. Such different types of server load can noticeably decrease its performance because of mixing different requirements for hardware. Because of this, many companies started to separate the application content by type through different servers worldwide. Eventually, new providers that turned this into a part of their business appeared on the market.

Content Delivery Network gives great performance boost because of its architecture design. In such network, many servers are situated worldwide in such a way that user receives static content from the geographically closest possible server. This technique makes it possible to achieve lower latency times and, in addition, to save costs on paying for the external traffic (if talking about large ISPs). Furthermore, CDN makes it possible to scale easier web applications to the larger sizes. That means that CDN offers greater stability for web applications in time of clients activity surge. For example, that could be a demand for a new version of software package, when many users want to download an updated installation file or a game archive.

Content delivery networks are based on many replica servers, which are situated in the geographical locations with high Internet traffic. Practically, the CDN works as follows:

1. User sends request to the *example.com* host to retrieve an HTML-page. At this stage, depending on the CDN configuration, static content can be downloaded from the origin server or already from the CDN edge server. If a CDN server does not have necessary resource copy, then it will download it and return to the user.
2. Next requests to the origin server (for static resources) will be redirected by BGP protocol to the CDN replica servers that will send a response. A dynamically generated content will still be retrieved from the origin host.

At this stage, BGP routers of user's provider are already aware of the best route paths to the edge servers, because they have received submissions with corresponding IP addresses from their neighbors. Sometimes, cached static data can be available to the user earlier that the source HTML-page. At this moment, we can affirm that CDN is an effective solution for companies, which need to return static content to a wide user base. The situation can be even more complicated, if there are many users on different continents.

3 Applied Method

The main purpose of this article is to verify two primary hypotheses:

1. The possibility of CDN usage depends on the number of external resources;
2. The CDN popularity depends on the number of Internet users in selected geographical region.

To verify these theses, a specialized application was implemented. This tool helps to analyze individual pages or a set of web pages from the perspective of external resources and CDN usage. Generated report consists of few web page parameters that help to tell how many internal and external resources are on the page. In addition, based on attributes analysis, the tool tries to detect the usage of CDN resources by specified web page. Usually, most of page resources (script files, media elements or CSS styles) are integrated into HTML document by using corresponding HTML tags. Except for the inline scripts and styles, all these assets have to include an appropriate attributes that can be parsed. By analyzing these attributes, it is possible to extract a collection of necessary HTML objects from the Document Object Model (DOM). Next, by comparing values of certain object properties (e.g. value of the *src* attribute) with the origin server URL, it is possible to say if the web page is using a resource sharing. Moreover, by comparing these values with the list of prepared known CDN URLs we can assume that the page use a CDN service.

The implemented tool was used to research a list of the 500 most popular hostnames that was based on The Moz Top 500 List [9]. The hostname inclusion in the list is based on its LRD (Linking Root Domains) index. For the web page, this index indicates the number of domains, which have links to this website. Higher LRD index demonstrates higher popularity. The Moz Top 500 list was generated based on 31 billion domain names [9]. As it will be revealed later in this article, the majority of domains are located in America, Europe and Asia. Thus, collected statistics are valid for these three regions. In addition, such situation indicates the regions where the Internet popularity and growth are at its highest level.

For every analyzed website, the implemented application returns a number of internal and external resources (except the resources that are loaded with JavaScript), an autonomous system number (AS number), IP address, location country and town (if available). This data helps to organize analyzed hostnames and to categorize them by geographic region. The list of CDN services was generated basing on the most popular and used providers of this type. The list that was used to identify CDN service is presented in Table 1. Presented CDN providers can be separated into two groups: public and private. That means only that they have different business models. Public CDN providers set up their price basing on the market average price. Private CDN providers work on the basis of private agreements, which doesn't mean that they are smaller than public services.

The data for the analysis was collected as follows: the list of the hostnames was analyzed a few times to eliminate corrupted values and supplement blank ones if they existed. Such data could have resulted from failed connections between the client (application) and the server. Repeated analysis also avoids questionable data that could have resulted from invalid parser results, which relies on rules described in HTML specification version 4.01 and 5.

Table 1. The list of CDN services that were used in comparison algorithm

CDN	Provider
cdn	–
akamai.net, akamaized.net, akamaiedge.net, akamaihd.net, akamaitechnologies.fr, edgesuite.net, edgekey.net, srip.net	Akamai Technologies
azureedge.net	Microsoft Azure
cloudfront.net	Amazon Cloudfront
cloudflare.net	Cloudflare Inc.
facebook.com, facebook.net, fbcdn.net	Facebook Inc.
fastly.net	Fastly
google, googlesyndication, doubleclick.net, googlehosted, gstatic, youtube	Google Inc.
llnwd.net	Limelight Technologies
twimg.com	Twitter Inc.
yimg, yahooapis	Yahoo Inc.
wp.com	WordPress

4 Implemented Application

For the purpose of this paper, a specialized tool, which analyzes web pages for external resources and CDN service usage, was built. The tool is called *CDNChecker*. It makes it possible to test a web page from the user's perspective, which is very important if we are talking about distributed services. The tool comes in a form of a lightweight executable file, which can be shared across the Internet to maximize data collection from different sources. In such way, we can obtain how user geographic location can affect the results. CDNChecker is built to minimize its influence on page download. Such architecture makes it possible to analyze offline HTML pages in situations when there is no available network access. In this case, some statistics such as AS number, IP address or hostname country are not available.

For the entered hostname, CDNChecker downloads the HTML page and analyzes it by the parser to distinguish appropriate object from the document object model. Next, with the help of lexical analyzer, the tool filters only that objects (script files, media elements, CSS styles), which have an appropriate attribute values. The grammar for this type of computation is defined in simple form and is presented below:

- For script files: HTML *script* tag with *type* or/and *src* attribute.
- For CSS styles: HTML *link* tag with *rel* or/and *href* attribute.

For images: HTML *img* tag with src attribute.

As a result, for a set of pages, application generates a report in CSV format, which contains a collection of strings. Each of these strings represents a record that characterizes specific page and includes a set of parameters:

- *hostname* - name of the target host;
- *scripts-count* - number of external script files;
- *styles-count* - number of external CSS styles;
- *img-count* - number of image objects;
- *is-using-cdn* - field that defines if the target hostname uses CDN;
- *cdn-references* - number of external resources, which are located on CDN service;
- *country* - the country of the hostname server location;

Statistics, which are presented in the next section, stand on such reports. After a few iterations (report generation for the entire list of hostnames), wrong (corrupted value) and blank values (unavailable host) were rejected and the number of hostnames reduced to 482 records.

5 Results

As were discussed in previous chapters, two main questions that stands in front of this research paper are related to the CDN usage detection and its correlation between number of external resources and host geographical position. Table 2 shows statistics about numerical distribution of occurrences of external resources (in brackets, the number of occurrences of both internal and external assets).

Table 2. Statistical spread of external resources

	Script files	Styling files	Images	The number of CDN references for all assets
Average	9.91 (16.33)	2.91 (4.22)	28.026	5.302
Minimal	0	0	0	0
Maximal	155	26	1179	188

From the data in Table 2, we can say that the majority of web pages try to use such optimizations to minimize web service load. Such statement is based on the number of external resources: nearly 60.7% of script files and 68.9% of graphics are external assets. Despite the continuous page size growth in the last years [10], web masters try to keep the performance of their services under control. Greater number of external resources means that the browser can download different assets from various servers in parallel [11].

To find the correlation between the number of external resources and CDN usage possibility, the data was separated into the four groups. The separation was based on the quartiles values. For every quartile, the average number of external resources and CDN usage was calculated. These data is presented in Table 3.

Table 3. Statistical spread of CDN usage by group

	Group bounds (min and max number of external resources)	Average number of external resources	Number of hosts in the group	Number of CDN usage occurrences	Percentage of hosts that use CDN
Group 1	<0, 9.75>	2.64	125	58	46.4%
Group 2	<9.75, 30>	20.57	136	67	49.3%
Group 3	<30, 58>	41.21	130	63	48.5%
Group 4	<58, 1258>	118.10	127	61	48%

From Table 3, we can see that CDN usage probability does not depend on the number of external resources. As was said earlier, the necessity of CDN is usually dictated by the server load and users geographic position, but not by the number of static resources. Next, Table 4 represents statistic data, which shows the numbers of web pages (among analyzed) that use CDN grouped by region.

From this table, we can see that CDNs are the most popular in American region – 51.7% of tested web pages use these services (North America and South America). In the European region, the situation is slightly different – only 44.68% of hosts use CDNs. In the Asian region, this number is 38.98% of all tested hosts. Such data demonstrates that there is a high demand for media content in American region (specifically in North America). Services such as YouTube, Netflix and Facebook have a high influence on content generation. Thus they use CDN (usually, built by themselves) to ensure the best possible experience for their users.

In both Europe and Asia, there is a smaller demand on CDN. This fact can be explained by smaller CDN infrastructure in these regions. Such situation may be

Table 4. CDN usage statistics across regions

	America	Asia	Australia	Europe	Oceania	Blank values	Total
CDN used	167 (51.70%)	23 (38.98%)	1 (50.00%)	42 (44.68%)	3 (75%)	0 (0%)	236
CDN not used	156 (48.30%)	36 (61.02%)	1 (50.00%)	52 (55.32%)	1 (25%)	18 (100%)	264
Number of Internet users	596 332 291	1 322 491 069	20 679 490	520 381 481	25 109 590	–	2 484 993 921
Total	323	59	2	94	4	18	500

caused by lower demand on such services or higher link quality, usually with lower price. In addition, received data makes it possible to say, that the number of Internet users in the region does not correlate with a high CDN popularity.

6 Conclusion and Future Work

Internet growth and evolution depends on user's needs. Different new usage scenarios appear. To make web applications faster, webmasters adjust their servers to perform better depending on the content type. Resource sharing and CDN services are combined to achieve maximum gain.

In this paper, the author has presented a brief overview of these technologies and provided a research on this topic. By summarizing the results, we can see that CDN services are used when there is a necessity to provide many users with static content. We observed that the number of Internet users in the region has not any considerable impact on CDN popularity either. Independently of the number of external resources, CDN is useful when there is high content demand on content of specific type. Static content such as page assets or media content are great candidates to be shared across edge servers.

The implemented application is on its first stage of development. The presented algorithm can be improved to dynamically detect loading resources, e.g. by JavaScript. Such heuristic methods can greatly improve overall detection quality to handle complicated HTML pages. After all, many companies can use CDNChecker to detect the configuration correctness from the user's perspective to achieve better results in network performance.

References

1. MDN, DOMContentLoaded – Event Reference. https://developer.mozilla.org/en/docs/Web/Events/DOMContentLoaded. Accessed 08 May 2017
2. MDN, load – Event Reference. https://developer.mozilla.org/en/docs/Web/Events/load. Accessed 08 May 2017

3. HTML Standard. https://html.spec.whatwg.org/multipage/scripting.html#script. Accessed 08 May 2017
4. Borzemski, L., Nowak, Z.: An empirical study of web quality: measuring the web from Wroclaw University of technology campus. In: Engineering Advanced Web Applications: Proceedings of Workshops in Connection with the 4th International Conference on Web Engineering (ICWE 2004), Munich, Germany (2004)
5. Mukaddim, P., Buyya, R.: A taxonomy and survey of content delivery networks, pp. 4–5. Springer, Heidelberg (2008)
6. Nordell, S.: Network latency - how low can you go? http://www.lightwaveonline.com/articles/print/volume-29/issue-6/feature/network-latency-how-low-can-you-go.html. Accessed 04 May 2017
7. West, M., Medley, J.: Content Security Policy. https://developers.google.com/web/fundamentals/security/csp/. Accessed 07 May 2017
8. West, M.: Content Security Policy Level 3. https://www.w3.org/TR/CSP/. Accessed 07 May 2017
9. Top Sites: The most important websites on the Internet. https://moz.com/top500. Accessed 11 May 2017
10. HTTP Archive – Trends. http://httparchive.org/trends.php?s=All&minlabel=Jan+1+2014&maxlabel=Dec+15+2016. Accessed 11 May 2017
11. Gourley, D., Totty, B., Sayer, M., Aggarwal, A., Reddy, S.: HTTP: The definitive guide, pp. 88–92. O'Reilly Media, Inc. (2002)

Parallelization of Selected Algorithms on Multi-core CPUs, a Cluster and in a Hybrid CPU+Xeon Phi Environment

Adam Krzywaniak(✉) and Paweł Czarnul

Faculty of Electronics, Telecommunications and Informatics,
Gdansk University of Technology, Gdansk, Poland
adam.krzywaniak@pg.gda.pl, pczarnul@eti.pg.gda.pl

Abstract. In the paper we present parallel implementations as well as execution times and speed-ups of three different algorithms run in various environments such as on a workstation with multi-core CPUs and a cluster. The parallel codes, implementing the master-slave model in C+MPI, differ in computation to communication ratios. The considered problems include: a genetic algorithm with various ratios of master processing time to communication and fitness evaluation times, matrix multiplication and numerical integration. We present how the codes scale in the aforementioned systems. For the numerical integration code that scales very well we also show performance in a hybrid CPU+Xeon Phi environment.

Keywords: Parallel programming · Multi-core CPU · Cluster · Intel Xeon Phi · Parallelization

1 Introduction

In the current high performance computing landscape, there are a variety of powerful compute devices that can be exploited with parallel programming. This includes multi-core CPUs such as Intel Xeons with typically 2 CPUs per workstation with up to 48 cores and 96 threads with Xeon E7-8894 v4 CPUs, up to 72 physical cores of the Xeon Phi x200 7290 processor. Apart from such CPUs, typically workstations and cluster nodes are equipped with GPUs. The latest NVIDIA V100 features 5120 CUDA cores and 16 GB HBM2 VRAM.

In terms of parallel programming paradigms, several can be distinguished including: master-slave, geometric parallelism, pipelining and divide-and-conquer. In this paper, we focus on three parallel master-slave applications that differ in compute-communication ratios and demonstrate how this affects speed-ups using various compute devices, including a multi-core CPU, a cluster and a hybrid CPU+Xeon Phi x100 coprocessor environment. This allows readers to predict speed-ups of other parallel applications with similar compute-communication ratios. Furthermore, it is a step towards building a precise model for such applications and systems in the MERPSYS execution time and energy simulation environment [1, 2].

The structure of the paper is as follows. Section 2 contains review of related works on parallelization of master-slave computations. Section 3.1 presents our testbed environments while Sects. 3.2, 3.3 and 3.4 our three representative parallel applications along with results of test runs performed in the aforementioned environments. Section 4 includes conclusions that summarize obtained results and link to the future work based on this paper – simulation of created applications in the MERPSYS environment designed for modeling and simulation of execution time and energy consumption [1, 2].

2 Related Work

There are several works that address parallelization and scheduling of master-slave processing schemes in high performance computing environments. These are possible with several programming APIs such as MPI, OpenMP and CUDA [3].

Paper [4] deals with off-line and on-line scheduling on heterogeneous master-slave platforms, including metrics such as total completion time, makespan, maximum response time. The authors have concluded that heuristics taking into consideration communication links' parameters offer best results. Experiments were conducted with MPI.

Work [5] presents very interesting theoretical modeling of evolutionary master-slave computing in terms of speed-ups for assumed both network and processing parameters. The authors present both theoretical and practical experiments showing almost linear speed-ups for around 100 slaves, however with an assumption that fitness evaluation in a slave requires 0.25 s/individual. The authors have used Distributed BEAGLE - their implementation of the master-slave architecture. The authors considered a Beowulf cluster made of homogeneous computers and a 100 Mbits/s Ethernet switch.

Paper [6] analyzes parallelization of a genetic algorithm for image restoration including various components that contribute to execution time: computations to be parallelized on the slave part T_np, sequential computations on the master T_nq as well as communication time T_nc. It has been concluded that observable parallelization can be seen on condition $T\_np \gg T\_nq + T\_nc$.

Paper [7] presents a Double-Layer Master-Slave Model (DMSM) that is targeted for distribution of independent tasks among cluster nodes (through MPI) and then processed within a node by a number of threads using OpenMP. The HPCVL developed DMSM library realized the proposed processing scheme in Fortran 90 and C. The authors presented that DMSM allows to reduce work imbalance of optimizations of the H2O2 molecule with fixed angles and varying bond lengths to 20% over 16 Sun T5140 nodes and presented benefits of their approach compared to MPI or OpenMP only solutions. It should be noted that recently, apart from the classic cluster systems with multi-core CPUs, more and more accelerators and coprocessors have been engaged for parallel processing. On one hand, such devices such as GPUs or Intel Xeon Phi x100 coprocessors (many-core CPUs in the latest x200 series) offer very good performance/Watt. On the other hand, exploiting full potential of such devices is not easy because it requires highly scalable code due to many more but less powerful cores compared to a standard multi-core CPU.

Paper [8] analyzes both performance and power consumption of several applications on an Intel Xeon Phi coprocessor, a SandyBridge Xeon CPU and Tesla C2050. The tests use the SHOC application benchmark. Tests for FFT, GEMM, MD and reduction show limits up to which the applications scale.

Paper [9] presents KernelHive – a framework for parallelization of computations and data - a set of independent data chunks and OpenCL processing kernels are distributed and scheduled across compute devices such as CPUs and GPUs in a cluster or even among clusters. The system is able to optimize kernel configurations including the numbers of groups and work items and take communication costs into consideration. The paper presents scalability of a parallel MD5 password-breaking application using brute force on a cluster with 16 nodes and a heterogeneous configuration with various CPUs and GPUs. A similar application is presented in [10] for encryption and decryption of large amounts of data using for various CPUs, GPUs and in a cluster environment, demonstrating good scalability for up to 4 nodes.

Paper [11] presents parallel computation of similarity measures between large vectors in a hybrid environment with multi-core CPUs and Intel Xeon Phi coprocessors. It is demonstrated how to best partition input data in such an environment, what the best numbers of threads per CPUs and Xeon Phis are as well as gain from overlapping communication and computations. It is shown that the implementation benefits from adding new compute devices in a heterogeneous environment.

Papers [12, 13] analyzed master-slave parallelization of matrix multiplication and numerical integration for various numbers of nodes and various data sizes.

3 Applications and Experiments

3.1 Testbed Environments

As two default testbed environments we used a workstation with two multi-core CPUs and a cluster of machines with multicore CPUs, both located at the Faculty of Electronics, Telecommunications and Informatics, Gdansk University of Technology, Poland.

The multi-core workstation includes two Intel® Xeon™ CPUs E5-2680 v2 with 10 2.80 GHz physical cores with HyperThreading, 25 MB Cache and 128 GB RAM. It is running Linux kernel version 2.6.32. We used Intel's implementation of MPI and launched 1 master and slaves on physical cores. In the case of one of the three master-slave applications analyzed and benchmarked in this paper we extended the aforementioned multi-core testbed environment with 2 Intel® Xeon Phi™ coprocessors which consist of 60 physical 1.052 GHz cores each with the possibility to use four logical processors per one core. We call this environment hybrid because of various performances of the CPUs and the Xeon Phis.

The cluster environment consists of 3 racks, each with 36 Intel® Xeon™ E5345 CPUs with 4 physical 2.33 GHz cores and HyperThreading, 4 MB cache and 8 GB RAM. Each node is running Linux kernel 2.6.32. For the cluster we used Open MPI. It should be noted that the performance of the workstation CPU is considerably higher than that of the cluster node CPU.

3.2 Parallel Genetic Algorithm

The first representative application is a parallel genetic algorithm in which the master is in charge of execution of successive iterations with successive populations. The master sends chromosomes for evaluation to slave processes. Slaves calculate the value of the fitness function and send the best individual back to the master. The master, based on results received from slaves, generates a new population and redistributes new individuals to slaves to repeat the procedure with a new generation. The application runs for given number of generations and returns the best achieved result.

Scalability of the solution will largely depend on the ratio of the time spent by the master (t_m) compared to the sum of the time of calculations performed by a slave (t_s) with the time of master-slave communication (t_c). In our initial Traveling Salesman Problem implementation using the genetic algorithm this ratio is large because fitness evaluation is just computing the path length. But it is easy to imagine that the length of the route between combinations of nodes (cities) is not the only factor that could be optimized. Other examples of fitness functions for the same set of data prepared by master (combination of nodes) could be evaluating e.g. difficulty level or cost of each path. Optimal conditions for different criteria for the same combination of nodes result in a complex optimization problem that was modeled by us in the paper. Because of that we tested a few ratios of (t_m/(t_c+t_s)) (calculated for 1 master+1 slave configuration). As a representative we have chosen solution of a problem for 50 nodes run for 500 generations of the genetic algorithm. The size of the population was set to 2000. Different ratios of (t_m/(t_c+t_s)) were obtained by multiple execution of fitness function during fitness evaluation in a slave node for one individual. Complexity of the optimization problem was modeled for 1, 100, 500 and 1000 loops of fitness function evaluation. Various ratios were marked respectively in the figures showing the tendency of

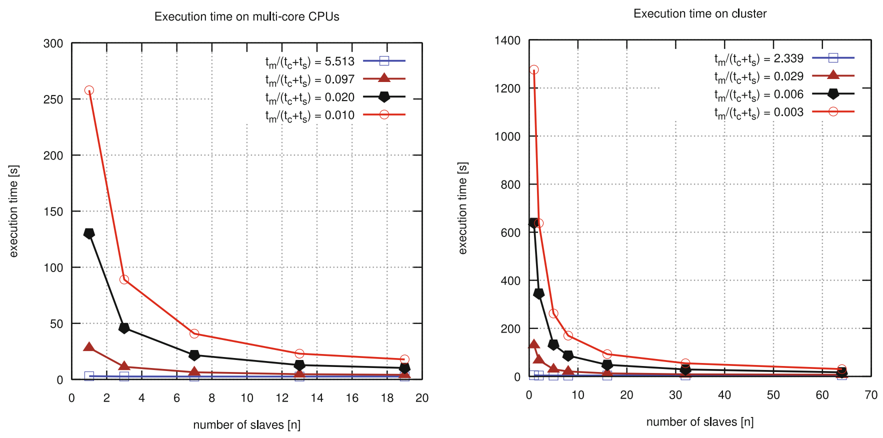


Fig. 1. Execution time of genetic algorithm application for various ratios of (t_m/(t_c+t_s)) for multi-core CPUs (left chart) and cluster (right chart) versus the number of slaves.

improving scalability and speed-up for decreasing (t_m/(t_c+t_s)) ratios. The application was run on both of the two default testbed environments mentioned in Sect. 3.1. Figure 1 presents execution times while Fig. 2 presents speed-ups.

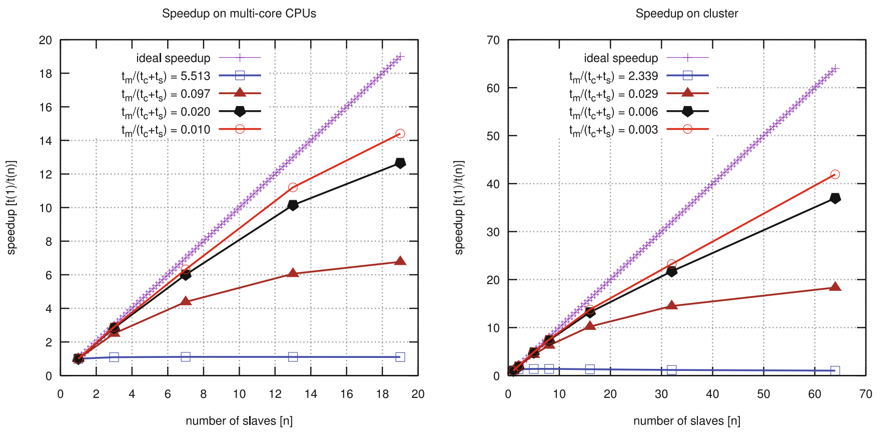


Fig. 2. Speedup of genetic algorithm parallel application for various ratios of (t_m/(t_c+t_s)) for multi-core CPUs (left chart) and on cluster (right chart) versus the number of slaves.

3.3 Parallel Algorithm for Matrix Multiplication

Matrix multiplication is the next problem that was considered by us in this paper in the context of parallelization in order to shorten the time of calculations. For the purpose of resolving this problem we have implemented another master-slave application. The algorithm we used assumes division of an output result matrix into sub-matrices and calculation of each sub-matrix using the standard algorithm. In our solution the master loads two multiplied input matrices and distributes the data required for each sub-matrix calculation among all the slaves. Slaves are designed to calculate sub-matrices (of a given size) of the result matrix. In order to be able to perform the calculations the master must prepare a data package containing proper columns and rows from two input matrices. A slave sends the result sub-matrix back to master and waits for the next calculation task. Therefore, the size of sub-matrix given to slaves to be calculated impacts strictly the size of data exchanged by a pair of master and slave in each iteration. That also – as in the previous genetic algorithm example, results in different communication to calculations ratios which affects the final scalability of the application.

For the purposes of this paper we chose multiplication of two square matrices of size 8000×8000. The size of the considered computational problem was determined by the least size of RAM available in considered testbed environments. The sizes of sub-matrices calculated by slaves varied during test runs starting from 800×800, through 400×400, 200×200 down to 100×100. Figure 3 presents execution times obtained during test runs for each sub-matrix size while Fig. 4 presents speedups. Both present the results for various numbers of slaves. The application was run in both testbed environments. The best scalability and execution times were obtained with the largest submatrices (800 × 800).

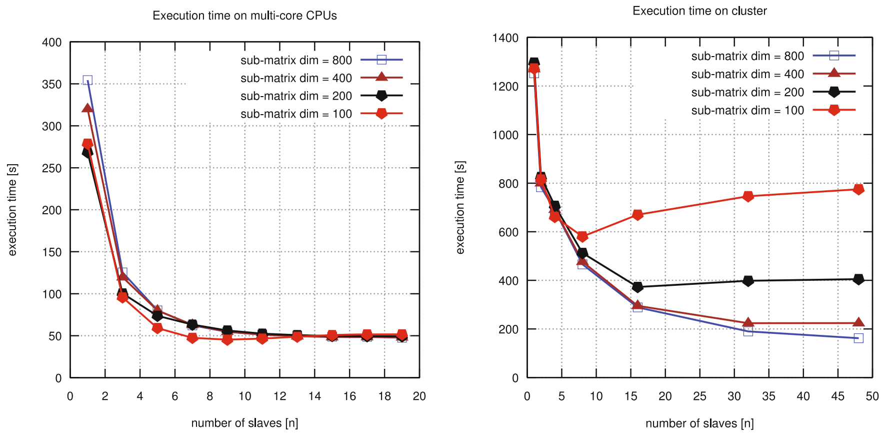


Fig. 3. Execution time for the matrix multiplication application in a multi-core testbed environment (left chart) and in a cluster (right chart) in relation to number of slaves.

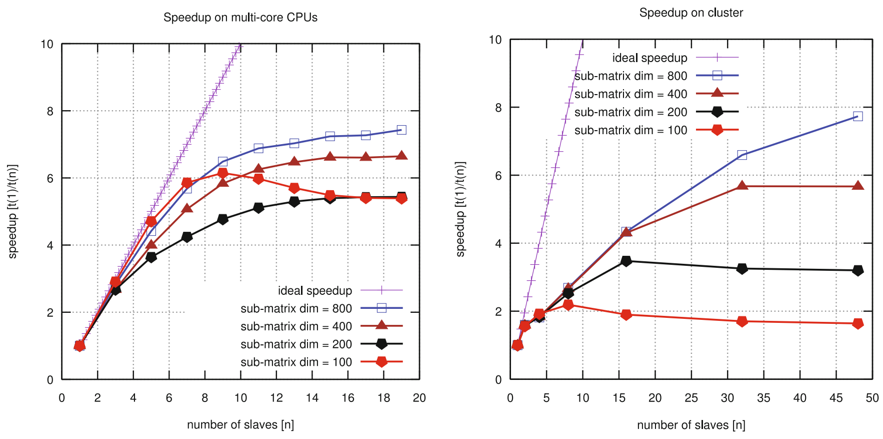


Fig. 4. Speedup for the parallel matrix multiplication application in a multi-core testbed environment (left chart) and in a cluster (right chart) in relation to number of slaves.

3.4 Parallel Algorithm of Integrate Calculation

The third considered problem is numerical integration. We implemented numerical integration in a master-slave paradigm as before, with a trapezoidal rule for computation of subranges. The master is responsible for distribution of subranges to be calculated by slaves as long as there are any left. Slaves perform integration of a given function for each subrange received from the master and send the result back. Complexity of computation blocks in slave nodes depends on the given accuracy of calculation set by PRECISION parameter. The tests were run for several values of the aforementioned parameter.

The results presented below were obtained for integration of the exp(sin(x)) function in the range of [−5,1500]. The PRECISION parameter during test runs was decreased 100 times per each run starting from the value 0.1 down to 0.0000001 and the size of subranges sent to the slaves was fixed and set to 1 in each case. Therefore, complexity of slaves' computation blocks was increased 100 times in each next test run while the communication and size of exchanged data remained the same. The communication to computation ratios were different again in each of the presented test runs.

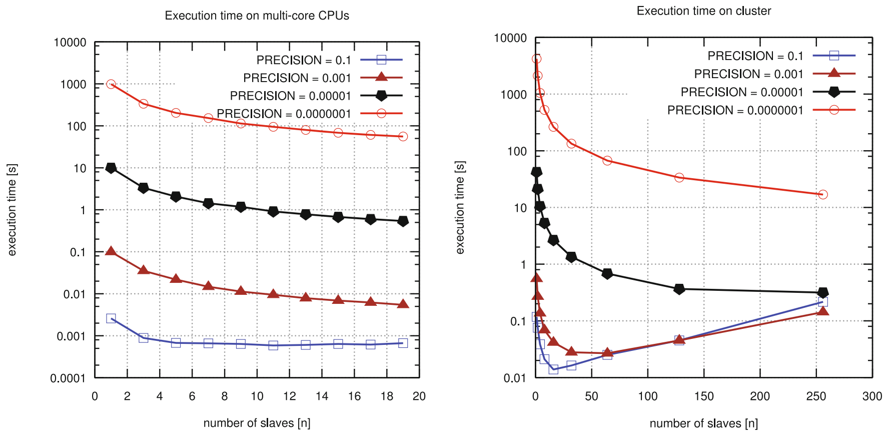


Fig. 5. Execution time for the integration application in a multi-core testbed environment (left chart) and in a cluster (right chart) in relation to number of slaves.

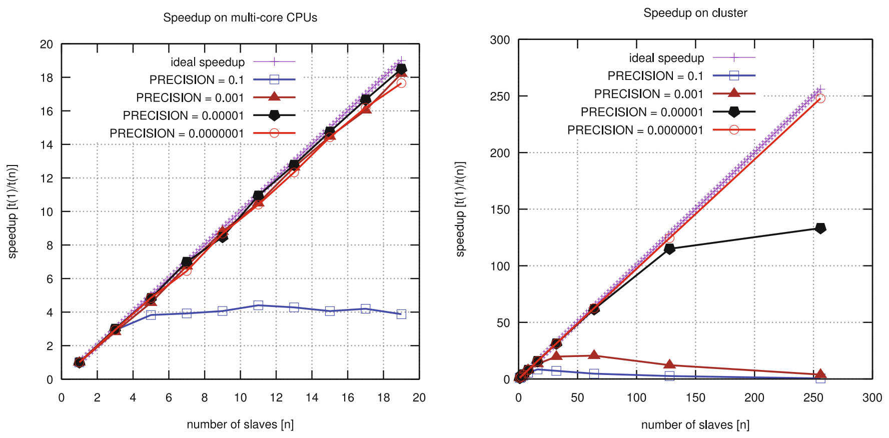


Fig. 6. Speedup for the integration in a multi-core testbed environment (left chart) and in a cluster (right chart) in relation to number of slaves.

The aforementioned application was run not only in both default testbed environments but also in a hybrid environment i.e. with two multi-core Intel Xeon CPUs with one or even two Intel Xeon Phi coprocessors added. The execution times and speedup values for the multi-core testbed environment and the cluster are presented in Figs. 5 and 6 respectively. The results allow to compare execution times as well as scalability between testbed environments also in the computation complexity context.

In the hybrid testbed environment we run the tests only for the best scaling case with the PRECISION parameter set to 0.0000001. The execution times and speedups for aforementioned hybrid testbed environment are presented in Figs. 7 and 8 respectively. It should be noted that the speed-up values were calculated against the performance of a single CPU core which is much more powerful than a single Phi core added for new slaves run on the Phi. Nevertheless we can see decrease in execution times and improvement of speed-ups when adding more slave threads that utilize cores of the first and the second Intel Xeon Phi coprocessor.

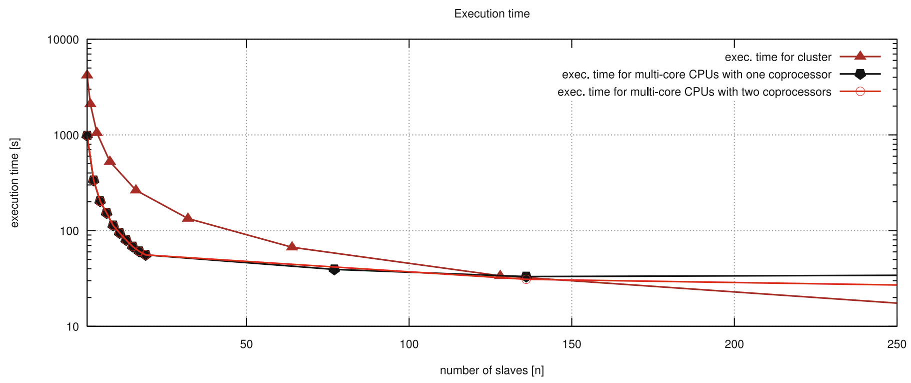


Fig. 7. Execution time for the integration application in the hybrid multi-core CPUs+Xeon Phi coprocessor(s) testbed environment in relation to number of slaves.

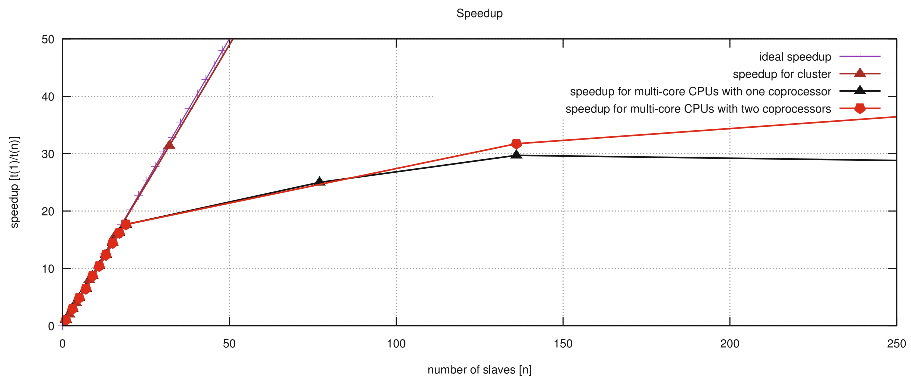


Fig. 8. Speedup for the integration application in the hybrid multi-core CPUs+Xeon Phi coprocessor(s) testbed environment in relation to number of slaves.

4 Summary and Future Work

In the paper we presented parallel implementations of three different C+MPI master-slave applications that differ in computation to communication ratios. The applications include a genetic algorithm, matrix multiplication and numerical integration. We ran experiments in two environments: a workstation with multicore CPUs and a cluster with nodes with multicore CPUs.

For the genetic algorithm as well as parallel matrix multiplication and numerical integration we presented conditions that are needed for obtaining good speed-ups in a parallel environment. Parallel matrix multiplication appeared to be a middle scaling application while the embarrassingly parallel numerical integration was scaling very well not only in a shared memory multicore CPUs machine and in a cluster but also in a hybrid CPUs+Xeon Phis environment.

Obtained speed-ups and growth for the genetic algorithm are in line with observations from [6], which suggests that good values are possible on condition that fitness evaluation is considerable compared to communication and synchronization costs in terms of time required. Speed-ups and their growth obtained for the matrix multiplication and numerical integration are similar to those obtained in [12] and in [13] respectively. In [13] adaptive integration generated subranges recursively based on a particular function. The subrange integration phase, however, is analogous to the one performed in this work. Differences in values stem from various startup times, bandwidths, synchronization costs, specific to the given environment.

We implemented the algorithms and obtained results as representative (in terms of various speed-ups) applications to be used next in our MERPSYS simulator for modeling and simulation of execution time and energy consumption [1, 2] for even greater sizes of input data and sizes of the environment. Another outcome of this work to readers is the possibility to assess potential scalability of such frequently used algorithms in various modern parallel environments.

References

1. Czarnul, P., Kuchta, J., Matuszek, M., Proficz, J., Rościszewski, P., Wójcik, M., Szymański, J.: MERPSYS: an environment for simulation of parallel application execution on large scale HPC systems. Simul. Model. Pract. Theor. **77**, 124–140 (2017). doi:10.1016/j.simpat.2017.05.009. Elsevier
2. Czarnul, P., Kuchta, J., Rościszewski, P., Proficz, J.: Modeling energy consumption of parallel applications. 2016 Federated Conference on Computer Science and Information Systems (FedCSIS), Gdansk, pp. 855–864 (2016)
3. Barlas, G.: Multicore and GPU Programming: An Integrated Approach. Morgan Kaufmann Publishers Inc., San Francisco (2014). ISBN: 9780124171404
4. Pineau, J.F., Robert, Y., Vivien, F.: Off-line and on-line scheduling on heterogeneous master-slave platforms. In: 14th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP 2006) (2006). doi:10.1109/PDP.2006.49
5. Dubreuil, M., Gagne, C., Parizeau, M.: Analysis of a master-slave architecture for distributed evolutionary computations. IEEE Trans. Syst. Man Cybern. Part B (Cybern.) **36**(1), 229–235 (2006). doi:10.1109/TSMCB.2005.856724

6. Chen, Y.-W., Nakao, Z., Fang, X.: Parallelization of a genetic algorithm for image restoration and its performance analysis. In: Proceedings of IEEE International Conference on Evolutionary Computation, Nagoya, pp. 463–468 (1996). doi:10.1109/ICEC.1996.542645
7. Liu, G., Schmider, H., Edgecombe, K.E.: A hybrid double-layer master-slave model for multicore-node clusters. J. Phys. Conf. Ser. **385**(1), 1–7 (2012)
8. Li, B., Chang, H.-C., Song, S., Su, C.-Y., Meyer, T., Mooring, J., Cameron, K.W.: The power-performance tradeoffs of the Intel Xeon Phi on HPC applications. In: Proceedings of the 2014 IEEE International Parallel & Distributed Processing Symposium Workshops (IPDPSW 2014), Washington, DC, USA, pp. 1448–1456. IEEE Computer Society (2014). doi:http://dx.doi.org/10.1109/IPDPSW.2014.162
9. Rościszewski, P., Czarnul, P., Lewandowski, R., Schally-Kacprzak, M.: KernelHive: a new workflow-based framework for multilevel high performance computing using clusters and workstations with CPUs and GPUs. Concurrency Comput. Pract. Exper. **28**, 2586–2607 (2016). doi:10.1002/cpe.3719
10. Niewiadomska Szynkiewicz, E., Marks, M., Jantura, J., Podbielski, M.: A hybrid CPU/GPU cluster for encryption and decryption of large amounts of data. J. Telecommun. Inf. Technol. **3**, 32–39 (2012)
11. Czarnul, P.: Benchmarking performance of a Hybrid Intel Xeon/Xeon Phi system for parallel computation of similarity measures between large vectors. Int. J. Parallel Program. **45**, 1091–1107 (2016). doi:10.1007/s10766-016-0455-0. Springer
12. Datti, A.A., Umar, H.A., Galadanci, J.: A beowulf cluster for teaching and learning. Procedia Comput. Sci. **70**, 62–68 (2015). doi:10.1016/j.procs.2015.10.034. ISSN: 1877-0509
13. Czarnul, P.: Parallelization of compute intensive applications into workflows based on services in BeesyCluster. Scalable Comput. Pract. Experience **12**(2), 227–238 (2011). ISSN: 1895-1767

E-Business Systems, Web Design, Mobile and Multimedia Systems

Verification of Web Traffic Burstiness and Self-similarity for Multiple Online Stores

Grażyna Suchacka(✉) and Alicja Dembczak

Institute of Mathematics and Computer Science,
University of Opole, Oleska 48, 45-052 Opole, Poland
gsuchacka@uni.opole.pl, alicja.dembczak@gmail.com

Abstract. Developing realistic Web traffic models is essential for a reliable Web server performance evaluation. Very significant Web traffic properties that have been identified so far include burstiness and self-similarity. Very few relevant studies have been devoted to e-commerce traffic, however. In this paper, we investigate burstiness and self-similarity factors for seven different online stores using their access log data. Our findings show that both features are present in all the analyzed e-commerce datasets. Furthermore, a strong correlation of the Hurst parameter with the average request arrival rate was discovered (0.94). Estimates of the Hurst parameter for the Web traffic in the online stores range from 0.6 for low traffic to 0.85 for heavy traffic.

Keywords: Web traffic · HTTP traffic · Web server · E-Commerce · Web store · Log analysis · Burstiness · Self-Similarity · Hurst parameter · Hurst index

1 Introduction

Numerous studies on the analysis and characterization of Internet traffic have confirmed its specific properties. Two very significant traffic features are its burstiness and self-similarity. *Burstiness* means that the traffic is highly variable, with traffic "bursts" observable on multiple time scales. The resulting time series, which is bursty on a wide range of time scales, may be statistically described as a *self-similar* process [1]. Burstiness and self-similarity have been identified both in the network traffic [2–5] and in Web server workloads [1, 6–9].

In reality, bursts in request arrival rates on the server may lead to transient server overloads and consequently, to a degraded server performance. Even a small amount of the traffic burstiness may degrade the server throughput [10, 11]. That is why this phenomenon has to be taken into account in Web server performance evaluation using the synthetic workload: to achieve reliable results of experiments testing the system performance, it is essential to model and generate bursty Web traffic [12–17].

In this paper we consider an arrival process of HTTP requests on Web servers which host B2C e-commerce websites, i.e., online stores. The motivation for our study was the fact that very few previous Web traffic analyses have been dedicated to e-business sites and the relevant literature lacks the comparative analysis of burstiness and self-similarity factors for multiple e-commerce environments. We obtained 24-hour

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 38th International Conference on Information Systems Architecture and Technology – ISAT 2017*, Advances in Intelligent Systems and Computing 655, DOI 10.1007/978-3-319-67220-5_28

access log data for seven various Web stores, differing in the type and size of the store offer, the website structure, and site popularity, and we used them to estimate burstiness and self-similarity factors for the e-commerce sites. To the best of our knowledge, there has not been such a wide study of the e-commerce traffic so far.

The rest of this paper is organized as follows. Section 2 explains a concept of self-similarity and discusses methods used to evaluate the traffic burstiness and self-similarity. Section 3 presents achieved results and Sect. 4 concludes the paper.

2 Approach for Evaluating Self-similarity and Burstiness of the Web Traffic

2.1 Definition of Self-similarity

Self-similarity may be defined in the context of the time series distribution [1]. Let $X = (X_t;\ t = 1, 2, \ldots)$ be a zero-mean, stationary time series. The m-aggregated series $X^{(m)} = (X_k^{(m)};\ k = 1, 2, \ldots)$ is defined by summing the time series X over nonoverlapping blocks of length m. Series X is *H-self-similar* if for all positive m, series $X^{(m)}$ has the same distribution as X rescaled by m^H:

$$X_t = m^{-H} \sum\nolimits_{i=(t-1)m+1}^{tm} X_i \tag{1}$$

for all $m \in N$. H-self-similar series X has the same autocorrelation function: $r(k) = E[(X_t - \mu)(X_{t+k} - \mu)]/\sigma^2$ as the series $X^{(m)}$ for all m.

The degree of self-similarity may be estimated by determining the *Hurst parameter* (*Hurst index*), denoted by H. For a self-similar series this parameter is higher than 0.5. The higher H is, the higher degree of self-similarity is revealed by the series.

Various statistical tests may be applied to assess the Hurst parameter. Popular tests operating in the time domain are the aggregate variance method and the R/S plot method. Other common tests, operating in the frequency domain, include the periodogram-based method, the wavelet-based estimator, and the Local Whittle estimator. Less common methods are multifractal analysis, detrended fluctuation analysis, and the Arby-Veitch estimator.

2.2 Estimation of the Hurst Parameter Using the Aggregate Variance Method

To verify the self-similarity of the traffic, we apply the aggregate variance method, which has been widely applied in previous Internet traffic analyses [1, 2, 5, 8, 9, 11, 13, 17, 18]. This test uses the fact that for a self-similar process variances of the sample mean are decaying more slowly than the reciprocal of the sample size [20].

Based on request timestamps read from log data a time series $X = (X_t;\ t = 1, 2, \ldots, N)$ is created, covering the time interval T. In our case each original series X has a duration of T = 24 h = 86400 s.

Value of m, i.e., duration of a subinterval, is given in seconds, $m \in [2, N/2]$. Consecutive values of m are generated as a geometric sequence 2^k, $k = 1, 2, \ldots,$

so that $m <= N/2$. The m-aggregated series $X^{(m)}$ are created for consecutive values of m and the variance of series $X^{(m)}$ is determined.

The variance of $X^{(m)}$ is then plotted against m on a log-log plot and approximated by a straight line by using the least squares method. The slope of the line $-\beta$ is computed and used to determine the Hurst parameter, given by:

$$H = 1 - \beta/2. \tag{2}$$

2.3 Estimation of the Burstiness Factor

For each analyzed e-commerce dataset the request arrival data is first plotted on many time scales to visually inspect the traffic burstiness. Then, a more rigorous analysis of a *burstiness factor* is performed in the following way [19].

Let L be the total number of requests that arrived on the Web server in the time interval T. Let λ be the average request arrival rate, given by:

$$\lambda = \frac{L}{T}. \tag{3}$$

Let the time interval T be divided into n equal subintervals of duration m. Let l_k be the number of requests that arrive in subinterval k and λ_k be the arrival rate of requests during subinterval k, given by:

$$\lambda_k = \frac{n}{T} \times l_k. \tag{4}$$

where $k = 1, 2, \ldots, n$. Let l^+ be the total number of requests that arrive in subintervals in which the subinterval arrival rate λ_k exceeds the average arrival rate λ. The *burstiness parameter* b_m is defined as the fraction of time during which the subinterval arrival rate exceeds the average arrival rate:

$$b_m = \frac{Number\ of\ subintervals\ for\ which\ \lambda_k > \lambda}{n}. \tag{5}$$

If the traffic is not bursty, it means that it is uniformly distributed over all subintervals and consequently, $b = 0$. On the other hand, for the bursty traffic $b > 0$.

For each analyzed e-commerce dataset we compute the burstiness parameter, b_m, for $m = 2, 4, 8, 16, 32, 64, 128$, and 256 s. We also determine the mean value of the burstiness parameter, b_{mean}, to compare the burstiness across the multiple datasets.

3 Results

3.1 Empirical Data Description

We analyzed access log data of e-commerce websites obtained from seven online retailers (the identities of the websites are not revealed for confidentiality restrictions).

The analyzed websites vary in the type and size of the store offer, the website structure and the traffic level in terms of the number of HTTP requests received in a 24-hour period. The highest traffic was registered for the online bookstore site (*SiteB*), offering books, films, and multimedia (89,486 requests in total) and for the website offering products and services for elderly people (*SiteE*, 70,352 requests). Two datasets were for the automotive branch e-stores: *SiteA1* (10,472 requests) and *SiteA2* (20,378 requests). Two other datasets, differing significantly in the numbers of samples, were for websites offering tourist equipment and clothes: *SiteT1* (2,616 requests) and *SiteT2* (37,819). The last dataset, *SiteH*, was for the site offering devices and systems for house equipment and contained only 7,666 samples (the corresponding website was not well positioned at that time).

General information on the analyzed Web stores' data is summarized in Table 1. Note that although dates of data collection differ between the individual websites, time samples in each dataset cover the total time interval of 24 h.

Table 1. Basic information on the analyzed e-commerce datasets.

	SiteB	*SiteE*	*SiteT2*	*SiteA2*	*SiteA1*	*SiteH*	*SiteT1*
Branch	Books	For elderly	Tourist	Auto-motive	Auto-motive	For house	Tourist
Date of data collection	Apr 1, 2014	Jan 25, 2016	Mar 29, 2015	Apr 3, 2015	Nov 12, 2016	Apr 10, 2015	Feb 24, 2015
Number of requests, L	89,486	70,352	37,819	20,378	10,472	7,666	2,616
Average request arrival rate, λ	1.04	0.81	0.44	0.24	0.12	0.09	0.03

3.2 Burstiness

Figures 1, 2, 3, 4 and 5 illustrate request arrival rates at different time scales (per subintervals of m = 4, 8, 16, 32, and 64 s) for the most numerous dataset, *SiteB*. Depending on the subinterval duration, data shown in the figures covers various observation windows. For example, Fig. 1 illustrates request arrival rates for 1200 4-second subintervals so the plotted data corresponds to an 80-minute time span. The higher the value of m is, the longer observation window is reflected in a figure.

Data in Fig. 5 corresponds to the whole one day (24 h). In this case a clear diurnal pattern of request arrivals is visible, with the least intensive traffic at night time, the gradually increasing traffic since 5 am till the peak traffic period starting at about 2 pm and lasting till about 10 pm.

A visual inspection of the plots confirm that the Web traffic arriving at *SiteB* is evidently bursty across several different time scales. Plots for other datasets are not presented in the paper due to space limits but they lead to similar conclusions on the traffic burstiness.

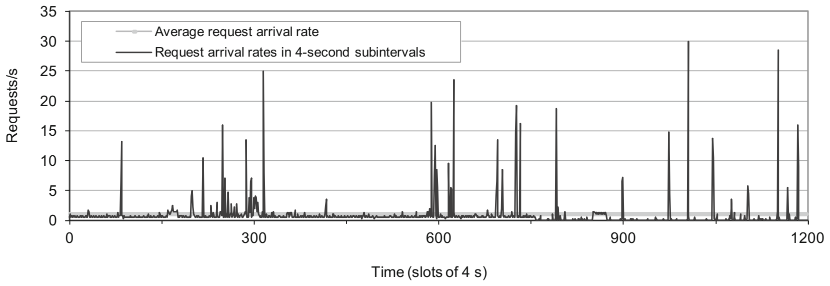


Fig. 1. Burstiness of the Web traffic on *SiteB* in slots of 4 s.

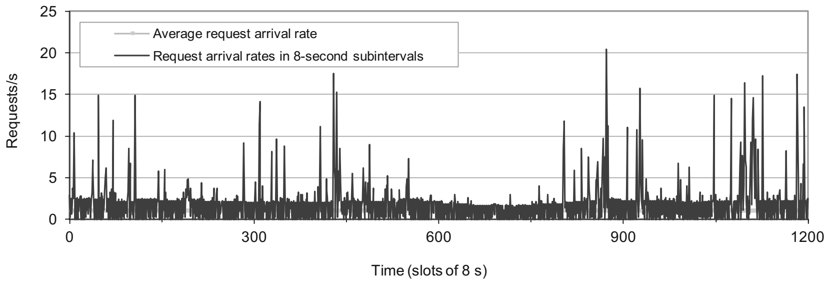


Fig. 2. Burstiness of the Web traffic on *SiteB* in slots of 8 s.

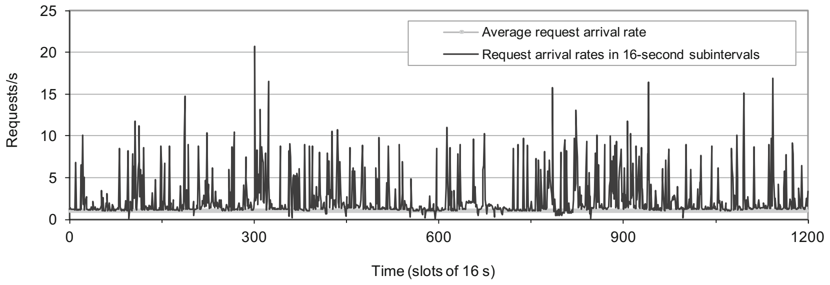


Fig. 3. Burstiness of the Web traffic on *SiteB* in slots of 16 s.

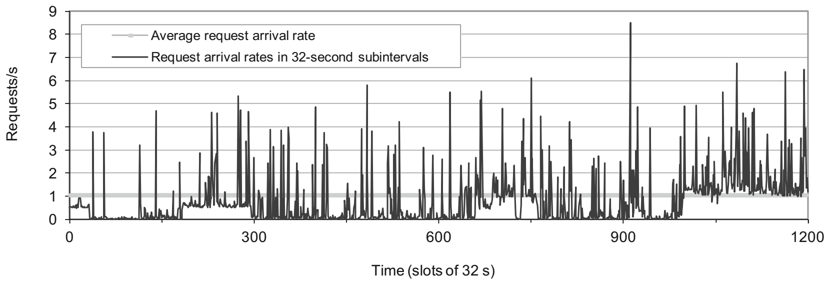


Fig. 4. Burstiness of the Web traffic on *SiteB* in slots of 32 s.

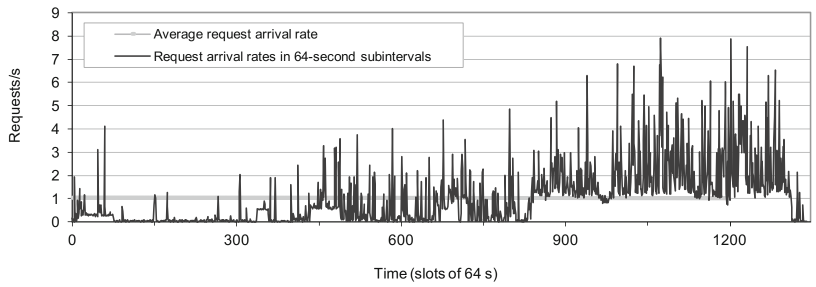


Fig. 5. Burstiness of the Web traffic on *SiteB* in slots of 64 s.

Burstiness visible on the plots of request arrival rates at different time scales is confirmed by estimates of the burstiness parameters, determined according to (5). Table 2 presents burstiness parameter values for different subinterval durations (2, 4, 8, 16, 32, 64, 128, and 256 s) and mean values of the burstiness parameter, denoted by b_{mean}. Figure 6 plots the burstiness parameter vs. subinterval duration for all the datasets. It can be seen that in general the burstiness factor tends to increase with the increase in the time scale. An exception from this tendency is the Web traffic registered for *SiteT1* and *SiteA2*.

Table 2. Burstiness factors for the analyzed datasets.

	SiteB	*SiteE*	*SiteT2*	*SiteA2*	*SiteA1*	*SiteH*	*SiteT1*
b_2	0.12	0.05	0.19	0.32	0.07	0.05	0.02
b_4	0.16	0.08	0.13	0.52	0.11	0.08	0.04
b_8	0.25	0.12	0.12	0.43	0.18	0.12	0.07
b_{16}	0.38	0.10	0.15	0.36	0.16	0.10	0.13
b_{32}	0.39	0.13	0.20	0.31	0.19	0.13	0.23
b_{64}	0.42	0.13	0.25	0.26	0.21	0.13	0.14
b_{128}	0.42	0.17	0.28	0.25	0.24	0.17	0.12
b_{256}	0.41	0.24	0.31	0.26	0.30	0.24	0.15
Mean burstiness (b_{mean})	0.32	0.13	0.20	0.34	0.18	0.13	0.11

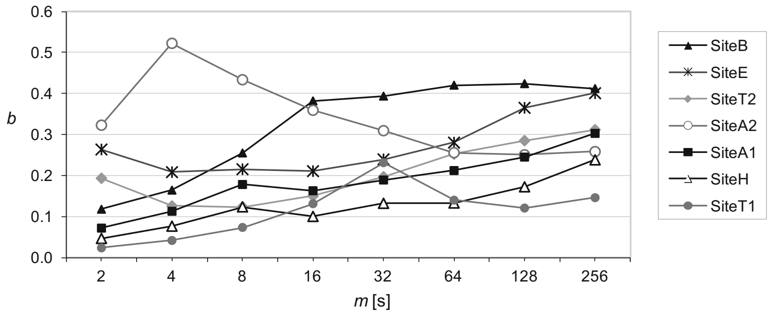


Fig. 6. Burstiness factor vs. subinterval duration.

3.3 Self-similarity

A visual inspection of the traffic burstiness and the determined burstiness factors suggest the presence of self-similarity in the Web traffic on all the analyzed e-commerce sites – even though the traffic stationarity at higher time scales is questionable. This is confirmed by estimates of H, all of which exceed 0.5 (Table 3).

Table 3. Hurst parameter determined for the analyzed datasets.

	SiteB	*SiteE*	*SiteT2*	*SiteA2*	*SiteA1*	*SiteH*	*SiteT1*
H	0.85	0.77	0.69	0.65	0.69	0.66	0.60

We examined the correlation between the H estimates and the burstiness parameters for different durations of a data aggregation subinterval, m. In the case of low values of m (2, 4, 8, 16, and 32) and b_{mean} there was no linear relationship or it was very weak (below 0.4). However, the relationship between H and b_{64} was moderate (0.64) and the relationships between H and b_{128} and b_{256} were quite strong (0.73 and 0.80, respectively).

Figures 7, 8, 9 and 10 show variance-time log-log plots for the analyzed time series. In all cases the shape of the line approximating the data significantly differs from –1, resulting in values of the Hurst parameter ranging from 0.6 to 0.85, depending on a dataset. These estimates confirm previous findings on H for e-commerce traffic, which was estimated as 0.66 for the traffic with the average arrival rate of 0.65 requests/s in the study [7] and ranged from 0.73 to 0.8, depending on the H estimating test, for the peak e-commerce traffic in the study [8].

We observed that higher traffic intensity levels on e-commerce sites correspond to higher values of the Hurst index. It is confirmed by a very high correlation between H and λ, equal to 0.94. This conclusion is also consistent with some previous findings for the non-e-commerce Web traffic [1, 6], stating that although self-similarity is not necessarily an invariant in all Web server workloads, it is evident in heavy workloads. However, in contrast, we identified the self-similarity in all the analyzed e-commerce server workloads, even for the websites subject to very low traffic levels.

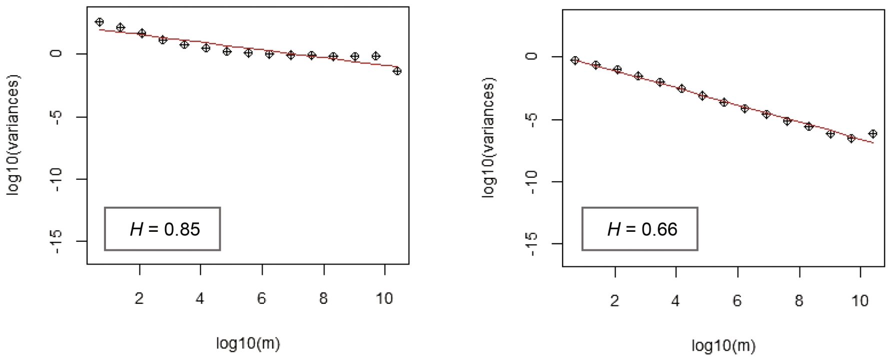


Fig. 7. Aggregate variance plot for *SiteB* (left) and *SiteH* (right).

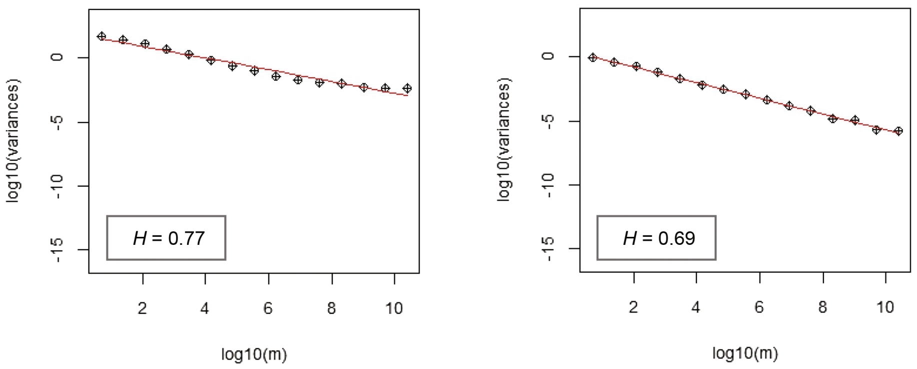


Fig. 8. Aggregate variance plot for *SiteE* (left) and *SiteA1* (right).

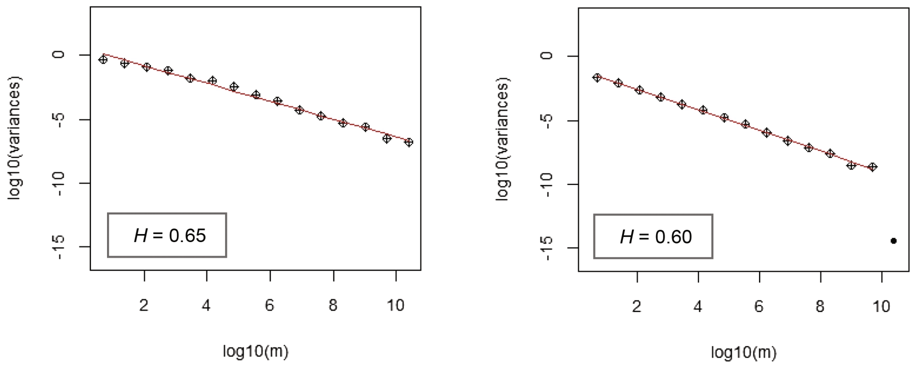


Fig. 9. Aggregate variance plot for *SiteA2* (left) and *SiteT1* (right).

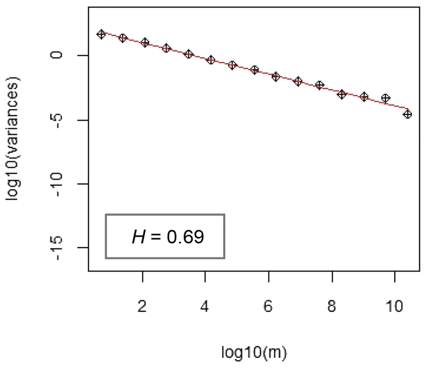


Fig. 10. Aggregate variance plot for *SiteT2*.

4 Concluding Remarks

In our study we verified burstiness and self-similarity of Web traffic on seven different e-commerce sites. The resulting estimates of the burstiness parameters (including mean burstiness factors ranging from 0.11 to 0.32, depending on a site) confirm the very variable character of the workloads on all the analyzed e-commerce servers.

Moreover, in all cases the Hurst parameter exceeds 0.5 which proves the presence of self-similarity in the e-commerce traffic (H estimates range from 0.6 for low traffic level to 0.85 for heavy traffic). Our results are consistent with older reports on H index for e-commerce severs with moderate [7] and heavy [8] traffic levels, estimated as 0.66 and 0.73-0.8, respectively. Our study also confirms some previous conclusions that a degree of self-similarity of Web traffic is a bit higher on e-commerce sites than on other sites [9].

In contrast to some previous related work for non-e-commerce Web traffic, we identified the self-similarity property in all seven analyzed e-commerce datasets, even for the websites subject to low traffic levels. Furthermore, we discovered that the more requests arrive at an e-commerce server, the higher degree of self-similarity is revealed by the traffic – there is a very strong correlation of the Hurst parameter with the average request arrival rate, equal to 0.94.

Our study advances the state-of-the-art on properties of the Web traffic on e-commerce servers. The use of several datasets for online stores differing in the offered products, the website structure, and the site popularity allows us to generalize the results to multiple e-commerce scenarios. Our findings may be useful in developing representative models of e-commerce workloads for Web server performance evaluation.

Acknowledgment. This paper is based upon work from COST Action IC1304 Autonomous Control for a Reliable Internet of Services (ACROSS), supported by COST (European Cooperation in Science and Technology).

References

1. Crovella, M., Bestavros, A.: Self-similarity in World Wide Web traffic: evidence and possible causes. ACM SIGMETRICS Perform. Eval. Rev. **24**(1), 160–169 (1996)
2. Park, C., Hernández-Campos, F., Le, L., Marron, J.S., Park, J., Pipiras, V., Smith, F.D., Smith, R.L., Trovero, M., Zhu, Z.: Long-Range dependence analysis of Internet traffic. J. Appl. Stat. **38**(7), 1407–1433 (2011)
3. Dymora, P., Mazurek, M., Strzałka, D.: Computer network traffic analysis with the use of statistical self-similarity factor. Annales UMCS Informatica AI **13**(2), 69–81 (2013)
4. Domańska, J., Domański, A., Czachórski, T.: A few investigations of long-range dependence in network traffic. In: Czachórski, T., Gelenbe, E., Lent, R. (eds.) ISCIS 2014, Information Sciences and Systems, part III, pp. 137–144. Springer, Cham (2014)
5. Olejnik, R.: Study of the character of APRS traffic in AX.25 network. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2013, CCIS, vol. 370, pp. 31–37. Springer, Heidelberg (2013)
6. Arlitt, M., Williamson, C.: Web server workload characterization: the search for invariants. ACM SIGMETRICS Perform. Eval. Rev. **24**(1), 126–137 (1996)

7. Vallamsetty, U., Kant, K., Mohapatra, P.: Characterization of e-commerce traffic. Electron. Commer. Res. **3**(1), 167–192 (2003)
8. Xia, C.H., Liu, Z., Squillante, M.S., Zhang, L., Malouch, N.: Web traffic modeling at finer time scales and performance implications. Perform. Eval. **61**(2–3), 181–201 (2005)
9. Suchacka, G., Domański, A.: Investigating long-range dependence in e-commerce Web traffic. In: Gaj, P., Kwiecień, A., Stera, P. (eds.) CN 2016, CCIS, vol. 608, pp. 42–51. Springer, Cham (2016)
10. Banga, G., Druschel, P.: Measuring the capacity of a Web server under realistic loads. World Wide Web **2**(1–2), 69–83 (1999)
11. Hernandez-Orallo, E., Vila-Carbo, J.: Analysis of self-similar workload on real-time systems. In: RTAS 2010, pp. 343–352. IEEE (2010)
12. Borzemski, L., Suchacka, G.: Web traffic modeling for e-commerce Web server system. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2009, CCIS, vol. 39, pp. 151–159. Springer, Heidelberg (2009)
13. Suchacka, G.: Generating bursty Web traffic for a B2C Web server. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2011, CCIS, vol. 160, pp. 183–190. Springer, Heidelberg (2011)
14. Lu, X., Yin, J., Chen, H., Zhao, X.: An approach for bursty and self-similar workload generation. In: Lin, X., Manolopoulos, Y., Srivastava, D., Huang, G. (eds.) WISE 2013, LNCS, vol. 8181, pp. 347–360. Springer, Heidelberg (2013)
15. Suchacka, G., Borzemski, L.: Simulation-based performance study of e-commerce Web server system – results for FIFO scheduling. In: Zgrzywa, A., Choroś, K., Siemiński, A. (eds.) Multimedia and Internet Systems: Theory and Practice, AISC, vol. 183, pp. 249–259. Springer, Heidelberg (2013)
16. Domańska, J., Domański, A., Czachórski, T.: Estimating the intensity of Long-Range Dependence in real and synthetic traffic traces. In: Gaj, P., Kwiecień, A., Stera, P. (eds.) CN 2015, CCIS, vol. 522, pp. 11–22. Springer, Cham (2015)
17. Jakóbik, A.: Big data security. In: Pop, F., Kołodziej, J., Di Martino, B. (eds.) Resource management for big data platforms. Algorithms, modelling, and high-performance computing techniques, Computer Communications and Networks, pp. 241–261. Springer, Cham (2016)
18. Gong, W.-B., Liu, Y., Misra, V., Towsley, D.: Self-similarity and long range dependence on the Internet: a second look at the evidence, origins and implications. Comput. Netw. **48**(3), 377–399 (2005)
19. Menascé, D.A., Almeida, V.: Capacity Planning for Web Services: Metrics, Models and Methods. Prentice Hall PTR, Upper Saddle River (2001)
20. Rose O.: Estimation of the Hurst parameter of Long-Range Dependent time series. Report No. 137. Institute of Computer Science, University of Wurzburg (1996)

Frequent Scene Change in Wavelet Video Compression

Andrzej Popławski(✉)

Electrical and Control Engineering, Faculty of Computer,
University of Zielona Góra, ul. Szafrana 2, 65-516 Zielona Góra, Poland
a.poplawski@imei.uz.zgora.pl

Abstract. Wavelet-based coders are noteworthy alternative to popular hybrid coders due to natural feature of scalability. By using a scalable wavelet coder it is possible to provide a video content for a wide range of customers who are using networks with different bitrates and devices with different screen sizes. In the paper results of experimental research of compression efficiency in the case of a frequently scene change was presented. For the need of experiments a new parameter SC was defined. The parameter indicates how often the scene change is performed. The scene change was simulated by changing the order of frames in video sequences. Effectiveness of video compression has been estimated by performing the series of experiments with a set of suitable prepared video sequences. Quality measurements were performed for the luminance component by calculating the mean value of the PSNR for all pictures in a given sequence. Table 1 presents the experiment results as PSNR values for SC parameter varies from 32 to 0 for various bitrates. The results show differences of the compression effectiveness depending on a frequency of scene change.

Keywords: Video coding · Wavelet transforms · Scene change · Video sequences · Scalability

1 Introduction

Nowadays compression of video sequences is commonly used technique to reduce size of a multimedia data. It is used in many fields both for transmission and storage the data. The compression is used in such areas as: stationary devices, mobile devices, in amateur and professional applications. It is applied by large number of users of mobile devices, for example smartphones, tablets, etc. In many cases, users are not aware that they are using advanced compression techniques. In recent years there are prepared a number of coding standards (e.g. H.264, H.265) that are based on hybrid coding techniques. Hybrid codecs are very popular and have a lot of advantages, however they have also disadvantages. One of them is an inconvenience in obtaining scalability. Scalability in hybrid codecs can of course be achieved by introducing into the data stream additional layers. This solution ensures scalability and the number of layers depends on the particular application. Nevertheless, usage of the scalable extension (Scalable Video Coding Extension) contributes to a slight drop in the coding efficiency compared to a situation where this extension is not used [1, 2].

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 38th International Conference on Information Systems Architecture and Technology – ISAT 2017*, Advances in Intelligent Systems and Computing 655, DOI 10.1007/978-3-319-67220-5_29

Another group of video codecs are wavelet video codecs. Many researchers are interested in wavelet video coding, mainly because of valuable features these types of codecs. The most important advantages are: fully embedded bitstream (it is possible to progressive decode with single coded data stream, SNR scalability), natural scalability in spatial domain (it is possible to get from single data stream correct video sequences at reduced spatial dimensions), natural scalability in time domain (it is possible to get from single data stream right video sequences at reduced frame frequency), facility to get exact bitrate (through to fully embedded bitstream).

Specified above advantages of wavelet codecs have special importance in such application as wireless systems or inhomogeneous networks [3, 4]. In these applications flow capacity can change in time. Usage fully scalable wavelet video codecs enables broadcasting of video content for wide group of users (e.g. users using networks with variable speeds, users using devices with different screen sizes).

Like in other video coding techniques, also in wavelet coders, similarities between adjacent frames are used in order to reduce redundancy. It is obvious that from time to time in a video sequence there is a scene change. Such situation can take place when recording camera has been changed. In this situation similarities between adjacent frames are generally small. That is why an encoder can't encode video material as effectively as in the case when scene change doesn't occur. As a matter of fact in the real footage a scene change does not occur frequently. However, the faster is camera movement, the more unlike are adjacent frames. Also quickly changing background in a movie e.g. rippling surface of water or flying confetti may cause significant dissimilarity between adjacent frames. In such cases we can often talk about scene change.

In the literature the issue of a scene change in a video sequence is mainly treated as a problem of detecting shots in the video. This is carried out for the purpose of segmentation and classification of video materials. The division of the sequence into shots is used, among others, for indexing and archiving of a video [5–7]. Amer et al. [8] shows a modified method of processing video sequences in case of a scene change. Xu et al. [9] proposes a new rate control algorithm based on variation of GOP in scene variation conditions. Poobalasingam et al. [10] proposes a strategy for GOP selection in HEVC, as well as classification of the studied sequences was performed, due to their dynamics, content and the occurrence of a scene change. In this paper a scene change is treated as case when sequence has more than one camera view and thus the scene changes. The above mentioned works mostly refer to hybrid encoders and focus on improving coding efficiency. However, there is no work presenting the decrease in compression efficiency in the case of frequent scene change and the impact of coded content on the compression quality. It is known that the scene change in a sequence have an adverse influence on compression efficiency. The question is how strongly the influence is and what effect the content has on the deterioration of the quality of the encoded images.

In the paper experimental results are presented on wavelet video coder under simulated condition of scene change. Research was carried out using a representative group of video test sequences. The selected sequences are commonly used to assess the effectiveness of video sequence compression. The video sequences were modified in order to obtain simulated scene change at predetermined frequencies.

2 Wavelet Video Coding

The key role in wavelet video coding plays time domain subband analysis with motion compensation (MCTF – *Motion Compensation Time Filtering*) [11–13]. Successive frames in a video sequence in most cases are very similar to each other, sometimes are almost the same. Reduction of a redundancy that exists between successive frames enables an elimination of unimportant information. The input pictures of a sequence are in first step grouped into groups of pictures (GOP) and processed jointly. The GOP size is based on number of levels of wavelet analysis in time domain. The size of the group of pictures is equal to *k*-th power of 2, where *k* is the number of levels of a wavelet temporal analysis [14].

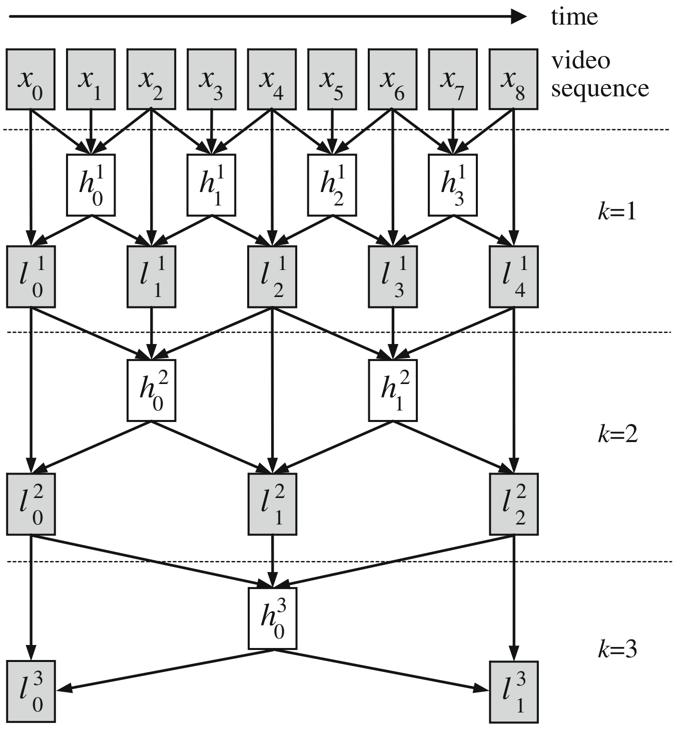

time
video sequence
x_0 x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8
h_0^1 h_1^1 h_2^1 h_3^1
k=1
l_0^1 l_1^1 l_2^1 l_3^1 l_4^1
h_0^2 h_1^2
k=2
l_0^2 l_1^2 l_2^2
h_0^3
k=3
l_0^3 l_1^3

Fig. 1. Three levels of temporal wavelet video analysis based on the LeGall 5/3 filters [14].

In the Fig. 1 a simplified temporal filtering scheme is presented for a group of eight pictures. The analysis starts at the first level of temporal decomposition (k = 1) and consists in filtering successive threesomes of input pictures x_{2t}, x_{2t+1}, x_{2t+2}. As a result, each of the three produces two filtered images: the low frequency component l_t and the high frequency component h_t (Fig. 1). At the next level of the temporal decomposition (k = 2), only low frequency components are processed, producing their own low frequency component l and the high frequency components h. The same scheme is repeated in the following decomposition levels, until only one low frequency and one high frequency component have been obtained. For a group of eight pictures, the final result is: one low frequency component and seven high frequency components. Typically, time domain analysis is performed for the five levels of temporal decomposition, which means that the GOP size is equal to 32 [14].

In the absence of similarities between successive images of video sequence (images from x_0 to x_8 in Fig. 1), this will adversely affect to effectiveness of the analysis in the time domain. Such state can cause termination of temporal analysis for example after the second stage of wavelet analysis in time domain ($k = 2$) because execution of next level of temporal decomposition may be inefficient.

In practice, a wavelet analysis is performed with the use of the so called lifting structure [15–17]. Lifting implementations of the temporal filters in wavelet video coders include motion-compensated prediction and update operations. Motion compensated temporal analysis is performed with the use of the lifting scheme [18] and based on the LeGall 5/3 filters [19]. Components produced in the process of motion compensated temporal filtering are in the next step analyzed in a two dimensional spatial domain [20]. The Daubechies 9/7 wavelet filters are used in this step of analysis [21, 22]. As a result, we obtain a three-dimensional wavelet analysis of the input video sequence (one dimension in time and two dimensions in 2D space) [11].

3 Methodology

The main aim of the experimental research was to examine behavior of a wavelet video coder in frequent scene change conditions. Occurrences of a scene change in video sequences imply a less effective analysis in time domain. Such situation is resulting from a lack of similarity between adjacent frames on the border of the scene change. This in turn can cause a significant redundancy decrease in an input video sequence. In research on wavelet video codecs that are available in a literature, there are used video sequences without scene change. The research are focused mainly on search for solutions that raise compression efficiency [23], reduce a coding delay which is a drawback of this codec [24], optimize coder parameters for special purposes [14] as well as use wavelet coders in medical applications [25].

For the experiments, six test video sequences were taken: Basket, City, Crew, Harbour, Ice and Soccer in format CIF 30 Hz (352 $\times$ 288) and with a length of 6.4 s (192 images). The sequences selected are commonly used to assess the effectiveness of video sequence compression. They are characterized by variable dynamic both in the foreground and in the background [14]. In the experiments, the MC-EZBC codec was used [26] with some modifications.

In the paper there was introduced SC parameter. The value of the SC parameter determines how often a scene change occurs in an input video sequence. It was assumed that in the research will be used test video sequences, for which the SC parameter will receive values: 32, 16, 8, 4, 2, 1. Accordingly, the least frequent scene change will take place every 32 images, next every 16 images, while the most frequent every one image. Thus, for the value of SC parameter equal to 1, every frame of a video sequence will be dissimilar to next or previous frame in the video sequence.

Furthermore, it was assumed that for the value of SC parameter equal to 0 the scene change doesn't occur. Video coding in this case will be performed with usage of the test video sequences without the scene change.

In the research it is very important to properly choose the material for experiments. The content of analyzed images for a given sequence should be the same for each value of the SC parameter. That is why it was decided to use original test video sequences in which order of images was changed. Test video sequences used in the research were modified in such a way as to contain the same contents regardless of assumed value of the SC parameter.

In order to simulate a scene change, the original test video sequence that consists of 192 pictures has been divided into two equal fragments. The first portion was successive pictures in the sequence from 1st to 96th, the second portion was successive pictures from 97th to 192nd (Fig. 2). In the next step, pictures from first and second portion were joined alternately. The number of successive combined pictures of each part depends on value of the SC parameter. For example, for a scene change every 4 pictures (SC = 4), in the resulting sequence were stored successive pictures from 1st to 4th, then pictures from 97th to 100th, afterwards from 5th to 8th, then 101st to 104th and so on. The successive pictures are taken from the original video sequence (Fig. 2). With such a method, every four pictures the simulated scene change occurs. Pictures 4th and 5th then 8th and 9th of the new sequence are dissimilar enough. We can thus say that the change of scene took place.

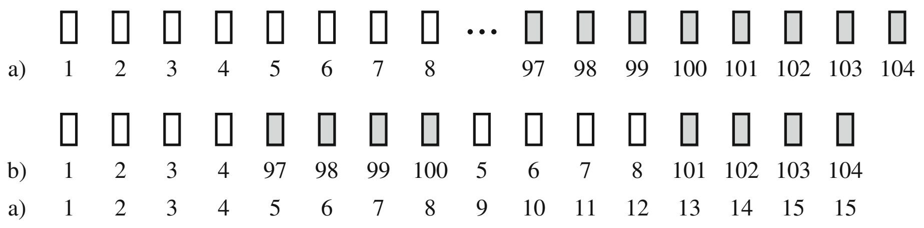

a) 1 2 3 4 5 6 7 8 ··· 97 98 99 100 101 102 103 104
b) 1 2 3 4 97 98 99 100 5 6 7 8 101 102 103 104
a) 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 15

Fig. 2. Reordering scheme of frames in the sequence for SC parameter equal to four; (a) image numbers of the original sequence, (b) image numbers of the modified sequence.

Such actions make that a nature of newly created sequence (dynamics, colors) is preserved. The new sequence was created with the same images, but only in different order of appearance. For each original test sequence there was created six modified video test sequences for values of the SC parameter equal to: 32, 16, 8, 4, 2, 1. For example, based on the original test sequence Basket, were created the following test sequences: Basket32, Basket16, Basket08, Basket04, Basket02, Basket01 (digits in names correspond to the value of the parameter SC). Therefore for the experiments, there were created in total 36 modified test video sequences.

4 Experimental Results

In this section of the paper results of experimental studies will be presented. The tests were executed on PC computer with Intel Core i7 2700K and 8 GB RAM. To measure the quality, the PSNR (*Peak Signal-to-Noise Ratio*) was used. The measurements were performed for the luminance component by calculating the mean value of the PSNR for all the pictures in a given sequence. This method is widely applied by many researchers. The experiments consisted in encoding and then decoding test video sequences with ten various assumed bitrates: 256, 384, 512, 640, 768, 896, 1024, 1152, 1280, 1408 kb/s. In total, 462 encoding/decoding operations (77 operations for each of the 6 video sequences tested) were performed.

A decoded sequence was compared to the original sequence, and the quality of the decoded sequence was measured. In the paper the Video Sequence PSNR Difference (VSPD) measure, defined as (1) is introduced:

$$VSPD = PSNR_0 - PSNR_{SC}[dB] \tag{1}$$

where:

$PSNR_0$	– the PSNR value in the event of the absence of a scene change,
$PSNR_{SC}$	– the PSNR value in the event of the presence of a scene change.

This measure defines the difference between the PSNR value in the absence of a scene change and the PSNR value in the presence of a scene change.

Table 1 shows the obtained values of the PSNR for particular SC values, for the tested video sequences and bitrates from 256 kb/s to 1408 kb/s. Table 2 shows the obtained values of the VSPD (PSNR decrease) for increasing frequency of scene change from 32 images to 1 image. The values are shown for all tested video sequences and for all bitrates. The biggest drop in the PSNR value was recorded for the City sequence – on average 5.44 dB, from 0.71 dB for SC = 32 and bitrate equal to 1280 kb/s to 11.78 dB for SC = 1 and bitrate equal to 1408 kb/s (Table 2). Whereas the smallest drop in the PSNR value was noted for the Crew sequence – on average 0.43 dB, most of 2.30 dB for SC = 1 and bitrate equal to 512 kb/s (Table 2). For the remaining video sequences the maximum decreases of the PSNR values were (Table 2):

- 6.07 dB at 1408 kb/s for the Basket sequence, on average 2.33 dB,
- 6.58 dB at 1408 kb/s for the Harbour sequence, on average 2.60 dB,
- 5.77 dB at 640 kb/s for the Ice sequence, on average 2.81 dB,
- 6.08 dB at 1408 kb/s for the Soccer sequence, on average 2.59 dB.

For all tested video sequences the VSPD values are the greater the more often the scene change occurs. The drop in the PSNR value is bigger for sequences with smooth movement and low dynamic (e.g. City) in contrast to sequences with high dynamic (e.g. Crew, Harbour). It is due the fact that for video sequences with high dynamic, differences between successive frames are smaller than in video sequences with lower amount of motion. Accordingly, the occurrence of a scene change in sequences with high dynamic cause a smaller decrease in PSNR compared to the sequences with less dynamics.

Table 1. PSNR values depending on SC parameter value and bitrate.

Sequence name	SC	PSNR [dB] at a given bitrate									
		256	**384**	**512**	**640**	**768**	**896**	**1024**	**1152**	**1280**	**1408**
Basket	**0**	12,79	24,91	26,56	27,89	28,88	29,66	30,50	31,15	31,75	32,34
	32	12,72	24,35	26,05	27,37	28,34	29,18	30,02	30,67	31,32	31,96
	16	12,69	23,89	25,66	26,96	27,93	28,79	29,54	30,28	30,91	31,48
	8	19,57	23,39	25,09	26,31	27,35	28,18	28,89	29,70	30,31	30,94
	4	20,44	22,87	24,33	25,50	26,42	27,25	28,01	28,69	29,45	29,98
	2	20,65	22,26	23,45	24,33	25,20	25,93	26,61	27,26	27,86	28,47
	1	20,40	21,48	22,31	23,06	23,67	24,33	24,83	25,34	25,83	26,27
City	**0**	34,51	36,66	38,01	39,04	39,84	40,37	41,10	41,63	42,00	42,47
	32	32,92	35,21	36,50	37,89	38,77	39,45	40,00	40,71	41,29	41,64
	16	31,72	34,04	35,66	36,90	37,91	38,70	39,38	39,94	40,60	41,14
	8	29,98	32,08	33,66	35,03	36,16	37,10	37,92	38,63	39,15	39,72
	4	28,28	29,86	31,29	32,35	33,33	34,37	35,21	35,94	36,69	37,33
	2	26,83	28,03	29,00	29,87	30,66	31,35	32,01	32,64	33,33	33,90
	1	25,65	26,56	27,24	27,85	28,40	28,98	29,46	29,90	30,31	30,69
Crew	**0**	31,63	33,13	34,32	35,11	35,86	36,60	37,11	37,68	38,17	38,56
	32	31,88	33,43	34,62	35,50	36,24	36,98	37,58	38,07	38,57	39,02
	16	31,71	33,26	34,46	35,35	36,07	36,86	37,44	37,98	38,45	38,93
	8	31,49	33,00	34,23	35,10	35,83	36,63	37,17	37,75	38,25	38,67
	4	31,17	32,69	33,91	34,78	35,57	36,28	36,95	37,54	37,97	38,49
	2	30,66	32,09	33,31	34,21	35,00	35,69	36,46	36,98	37,51	37,98
	1	29,73	31,00	32,02	32,95	33,69	34,37	34,99	35,62	36,22	36,65
Harbour	**0**	28,46	29,98	31,08	31,95	32,68	33,30	33,86	34,53	34,94	35,50
	32	27,82	29,43	30,45	31,44	32,22	32,88	33,39	34,04	34,52	35,05
	16	26,94	28,64	29,68	30,68	31,47	32,15	32,78	33,28	33,99	34,39
	8	26,64	28,21	29,20	30,16	30,93	31,67	32,28	32,80	33,27	33,89
	4	26,29	27,71	28,66	29,52	30,26	30,88	31,48	32,10	32,56	33,04
	2	25,17	26,90	27,89	28,59	29,35	29,92	30,47	30,98	31,44	31,88
	1	22,05	23,76	24,79	25,60	26,26	26,89	27,46	27,98	28,56	28,92
Ice	**0**	35,09	37,35	38,85	40,37	41,43	42,41	43,45	44,29	44,95	45,60
	32	33,99	36,28	37,85	39,33	40,43	41,42	42,34	43,23	44,00	44,68
	16	33,43	35,74	37,30	38,81	39,96	40,93	41,74	42,79	43,58	44,17
	8	32,86	35,15	36,75	38,18	39,43	40,40	41,26	42,25	42,94	43,68
	4	32,28	34,45	36,03	37,26	38,58	39,57	40,41	41,22	42,21	42,81
	2	31,58	33,48	35,19	36,43	37,53	38,70	39,61	40,40	41,17	41,96
	1	29,83	31,78	33,23	34,60	35,72	36,75	37,83	38,72	39,62	40,41
Soccer	**0**	30,98	32,81	34,25	35,41	36,33	37,27	38,04	38,75	39,38	39,89
	32	30,20	31,82	33,16	34,15	35,09	36,01	36,77	37,37	37,98	38,59
	16	30,00	31,64	32,97	33,99	34,88	35,80	36,50	37,16	37,80	38,39
	8	29,70	31,24	32,47	33,45	34,32	35,03	35,89	36,56	37,12	37,71
	4	29,40	30,75	31,93	32,79	33,60	34,34	35,05	35,69	36,27	36,79
	2	29,01	30,25	31,16	31,99	32,65	33,28	33,90	34,43	35,02	35,43
	1	28,43	29,52	30,28	31,02	31,58	32,07	32,53	32,99	33,42	33,81

Table 2. VSPD values depending on SC parameter value and bitrate.

Sequence name	SC	VSPD [dB] at a given bitrate									
		256	**384**	**512**	**640**	**768**	**896**	**1024**	**1152**	**1280**	**1408**
Basket	**32**	–	0,56	0,51	0,52	0,54	0,48	0,48	0,48	0,43	0,38
	16	–	1,02	0,90	0,93	0,95	0,87	0,96	0,87	0,84	0,86
	8	–	1,52	1,47	1,58	1,53	1,48	1,61	1,45	1,44	1,40
	4	–	2,04	2,23	2,39	2,46	2,41	2,49	2,46	2,30	2,36
	2	–	2,65	3,11	3,56	3,68	3,73	3,89	3,89	3,89	3,87
	1	–	3,43	4,25	4,83	5,21	5,33	5,67	5,81	5,92	6,07
City	**32**	1,59	1,45	1,51	1,15	1,07	0,92	1,10	0,92	0,71	0,83
	16	2,79	2,62	2,35	2,14	1,93	1,67	1,72	1,69	1,40	1,33
	8	4,53	4,58	4,35	4,01	3,68	3,27	3,18	3,00	2,85	2,75
	4	6,23	6,80	6,72	6,69	6,51	6,00	5,89	5,69	5,31	5,14
	2	7,68	8,63	9,01	9,17	9,18	9,02	9,09	8,99	8,67	8,57
	1	8,86	10,10	10,77	11,19	11,44	11,39	11,64	11,73	11,69	11,78
Crew	**32**	−0,25	−0,30	−0,30	−0,39	−0,38	−0,38	−0,47	−0,39	−0,40	−0,46
	16	−0,08	−0,13	−0,14	−0,24	−0,21	−0,26	−0,33	−0,30	−0,28	−0,37
	8	0,14	0,13	0,09	0,01	0,03	−0,03	−0,06	−0,07	−0,08	−0,11
	4	0,46	0,44	0,41	0,33	0,29	0,32	0,16	0,14	0,20	0,07
	2	0,97	1,04	1,01	0,90	0,86	0,91	0,65	0,70	0,66	0,58
	1	1,90	2,13	2,30	2,16	2,17	2,23	2,12	2,06	1,95	1,91
Harbour	**32**	0,64	0,55	0,63	0,51	0,46	0,42	0,47	0,49	0,42	0,45
	16	1,52	1,34	1,40	1,27	1,21	1,15	1,08	1,25	0,95	1,11
	8	1,82	1,77	1,88	1,79	1,75	1,63	1,58	1,73	1,67	1,61
	4	2,17	2,27	2,42	2,43	2,42	2,42	2,38	2,43	2,38	2,46
	2	3,29	3,08	3,19	3,36	3,33	3,38	3,39	3,55	3,50	3,62
	1	6,41	6,22	6,29	6,35	6,42	6,41	6,40	6,55	6,38	6,58
Ice	**32**	1,10	1,07	1,00	1,04	1,00	0,99	1,11	1,06	0,95	0,92
	16	1,66	1,61	1,55	1,56	1,47	1,48	1,71	1,50	1,37	1,43
	8	2,23	2,20	2,10	2,19	2,00	2,01	2,19	2,04	2,01	1,92
	4	2,81	2,90	2,82	3,11	2,85	2,84	3,04	3,07	2,74	2,79
	2	3,51	3,87	3,66	3,94	3,90	3,71	3,84	3,89	3,78	3,64
	1	5,26	5,57	5,62	5,77	5,71	5,66	5,62	5,57	5,33	5,19
Soccer	**32**	0,78	0,99	1,09	1,26	1,24	1,26	1,27	1,38	1,40	1,30
	16	0,98	1,17	1,28	1,42	1,45	1,47	1,54	1,59	1,58	1,50
	8	1,28	1,57	1,78	1,96	2,01	2,24	2,15	2,19	2,26	2,18
	4	1,58	2,06	2,32	2,62	2,73	2,93	2,99	3,06	3,11	3,10
	2	1,97	2,56	3,09	3,42	3,68	3,99	4,14	4,32	4,36	4,46
	1	2,55	3,29	3,97	4,39	4,75	5,20	5,51	5,76	5,96	6,08

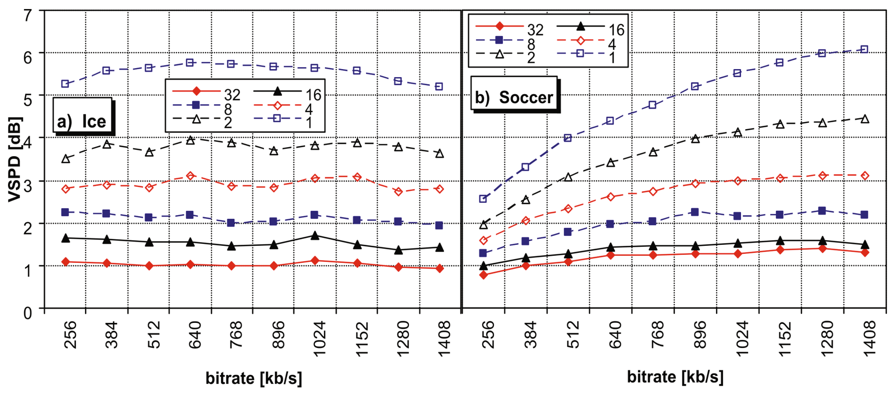


Fig. 3. The VSPD values for the testing sequences (a) Ice and (b) Soccer for SC parameter equal to: 32, 16, 8, 4, 2, 1.

It is noteworthy that for the City sequence for SC equal to 32 and 16 (partly also for SC = 8) there was observed negative value of the VSPD. This means that in these cases, occurrence of scene change causes an increase the PSNR value compared to a situation when there is no scene change. Additionally, for the Basket sequence in three cases for bitrate equal to 256 kb/s there was obtained very low level of PSNR values, around 12,7 dB (Table 1). These sequences were not properly decoded due to too low bitrate budget. Therefore, for this sequence the VSPD for bitrate 256 kb/s were not calculated (Table 2).

The presented results show, that in most cases, change of bitrate for the given SC value does not cause a significant change of the PSNR value. The VSPD value remains at a similar level (Fig. 3a). Only in few cases the value of the VSPD depends clearly on the bitrate. This is the case for the sequence Basket for SC = 1 and SC = 2, and for the sequence Soccer for SC = 1, 2, 4, 8 (Fig. 3b). It follows that the transmission speed has a relatively small influence on the VSPD value at particular values of the SC.

5 Conclusion

In the paper, results of experimental research on three-dimensional wavelet video codec under simulated condition of scene change are presented. For the experiments, video sequences were modified in order to obtain simulated scene change at predetermined frequencies equal to: 1, 2, 4, 8, 16 and 32 pictures. The selected sequences are commonly used to assess the effectiveness of video sequence compression. The results obtained in case of the presence of scene change were compared to the results, where there is no scene change. The research shows that a scene change in video sequences leads to drop in the PSNR value. The more frequent is the scene change, the bigger is the drop of the PSNR. Moreover, the less dynamic is a video sequence, the greater is decrease of the PSNR value. For sequences with very high dynamics, the drop of the PSNR values was almost five times smaller than the drop of the PSNR values for sequences with low dynamics.

References

1. Schwarz, H., Marpe, D., Wiegand, T.: Overview of the scalable video coding extension of the H. 264/AVC standard. IEEE Trans. Circuits Syst. Video Technol. **17**(9), 1103–1120 (2007)
2. Schwarz, H., Wien, M.: The scalable video coding extension of the H. 264/AVC standard. IEEE Sig. Process. Mag. **25**(2), 135 (2008)
3. Wu, D., Hou, Y.T., Zhang, Y.Q.: Scalable video transport over wireless IP networks. In: The 11th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, vol. 2, pp. 1185–1191 (2000)
4. Wu, D., Hou, Y., Zhang, Y.: Scalable video coding and transport over broadband wireless networks. Proc. IEEE **89**(1), 6–20 (2001)
5. Del Fabro, M., Böszörmenyi, L.: State-of-the-art and future challenges in video scene detection: a survey. Multimedia Syst. **19**(5), 427–454 (2013)
6. Eom, Y., Park, S., Yoo, S., Choi, J.S., Cho, S.: An analysis of scene change detection in HEVC bitstream. In: IEEE International Conference on Semantic Computing, pp. 470–474 (2015)
7. Eom, Y., Park, S., Chung, C.W.: An analysis of scene change detection using HEVC coding additional information. J. Broadcast Eng. **20**(6), 871–879 (2015)
8. Amer, H., Yang, E.H.: Scene-based low delay HEVC encoding framework based on transparent composite modeling. In: IEEE International Conference on Image Processing, pp. 809–813 (2016)
9. Xu, S., Yu, M., Fang, S., Peng, Z., Wang, X., Jiang, G.: New rate control optimization algorithm for HEVC aiming at discontinuous scene. WSEAS Trans. Comput. **14**, 598–606 (2015)
10. Poobalasingam, V., Izquierdo, E.: Steadiness analysis for optimal GOP size selection in HEVC. In: IEEE International Conference on Image Processing, pp. 799–803 (2016)
11. Choi, S.J., Woods, J.W.: Motion-compensated 3D subband coding of video. IEEE Trans. Image Process. **8**(2), 155–167 (1999)
12. Chen, P.: Fully scalable subband/wavelet coding. Doctoral Thesis (2003)
13. Misiti, M., Misiti, Y., Oppenheim, G., Poggi, J.M.: Wavelets and their Applications. Wiley, Hoboken (2013)
14. Popławski, A.: Selection of wavelet video codec parameters to optimize coding time. Int. J. Electron. Telecommun. **59**(4), 341–349 (2013)
15. Sweldens, W.: The lifting scheme: a custom-design construction of biorthogonal wavelets. Appl. Comput. Harmonic Anal. **3**(2), 186–200 (1996)
16. Daubechies, I., Sweldens, W.: Factoring wavelet transforms into lifting steps. J. Fourier Anal. Appl. **4**(3), 247–269 (1998)
17. Claypoole, R.L., Davis, G.M., Sweldens, W., Baraniuk, R.G.: Nonlinear wavelet transforms for image coding via lifting. IEEE Trans. Image Process. **12**(12), 1449–1459 (2003)
18. Pesquet-Popescu, B., Bottreau, V.: Three-dimensional lifting schemes for motion compensated video compression. IEEE Int. Conf. Acoust. Speech, Sig. Process. **3**, 1793–1796 (2001)
19. LeGall, D., Tabatabai, A.: Subband coding of digital images using symmetric short kernel filters and arithmetic coding techniques. Proc. Int. Conf. Acoust. Speech Sig. Process. IEEE **2**, 761–764 (1988)
20. Woods, J.W.: Subband Image Coding. Kluwer, Boston (1991)
21. Antonini, M., Barlaud, M., Mathieu, P., Daubechies, I.: Image coding using wavelet transform. IEEE Trans. Image Process. **1**(2), 205–220 (1992)

22. Cohen, A., Daubechies, I., Feauveau, J.C.: Biorthogonal bases of compactly supported wavelets. Commun. Pure Appl. Math. **45**(5), 485–560 (1992)
23. Pau, G., Pesquet-Popescu, B., van der Schaar, M., Vieron, J., Viéron, J.: Delay-performance trade-offs in motion-compensated scalable subband video compression. In: Proceedings of Advanced Concepts for Intelligent Vision Systems (2004)
24. Popławski, A., Domański, M.: Optimization of low delay wavelet video codecs. In: Proceedings of Picture Coding Symposium (2007)
25. Sriraam, N., Shyamsunder, R.: 3-D medical image compression using 3-D wavelet coders. Digit. Signal Proc. **21**(1), 100–109 (2011)
26. Rusert, T., Hanke, K., Wien, M.: Optimization for locally adaptive MCTF based on 5/3 lifting. In: Proceedings of Picture Coding Symposium, pp. 210–220 (2004)

A Versatile Hardware and Software Toolset for Computer Aided Inspection Planning of Machine Vision Applications

Stephan Irgenfried[1(✉)], Heinz Wörn[1], Stephan Bergmann[2], Mahsa Mohammadikajii[3], Jürgen Beyerer[3,4], and Carsten Dachsbacher[2]

[1] Karlsruhe Institute of Technology KIT, Institute of Anthropomatics and Robotics (IAR), Intelligent Process Control and Robotics (IPR), Engler-Bunte-Ring 8, 76131 Karlsruhe, Germany
{stephan.irgenfried,woern}@kit.edu

[2] Computer Graphics Group, Karlsruhe Institute of Technology KIT, Institute of Visualization and Data Analysis (IVD), Am Fasanengarten 5, 76131 Karlsruhe, Germany
{stephan.bergmann,dachsbacher}@kit.edu

[3] Vision and Fusion Laboratory (IES), Karlsruhe Institute of Technology KIT, Institute of Anthropomatics and Robotics (IAR), Adenauerring 4, 76131 Karlsruhe, Germany
mahsa.mohammadikaji@kit.edu, juergen.beyerer@iosb.fraunhofer.de

[4] Fraunhofer Institute of Optronics, System Technologies and Image Exploitation, Fraunhoferstraße 1, 76131 Karlsruhe, Germany

Abstract. The digital workflow and the set of tools described in this article support machine vision engineering during specification, design and implementation in numerous ways. In addition to the core idea of cad based vision system development, a generic data container format, simulation tools for real-time and photorealistic rendering using measured material BRDF data, and a MATLAB/ROS-controlled two-lightweight robot setup for experimental verification of simulation results are presented as parts of a semi-automatic planning process for a laser triangulation system. The planning process includes the annotation of CAD data with GD&T tolerance information using the ISO/PWI 10303-238 (STEP-NC) standard, initial system configuration by a machine vision expert, image formation simulation, system performance evaluation based on metrics applied on synthetic images and the capturing of real images with camera and laser light source, each mounted to a 7-axis KUKA LWR IV lightweight robot. The economic benefits of time and cost reduction of machine vision system planning are discussed as well as drawbacks and limits of the suggested workflow and tools.

Keywords: Machine vision · Optical inspection · CAD based vision · CAD based inspection planning · Sensor planning · Virtual vision · Simulation

1 Introduction

Engineering a machine-vision system can be a technically challenging and resource intensive process [1]. Many different inspection methods, camera, illumination, low- and high-level image processing algorithms, system and algorithm parameters span a large design space and the system engineer must pick and configure a possible solution to address the requirements of application and process. Computer aided inspection planning (CAIP), already being good practice in purely mechanical touch probing inspection systems [2], e.g. using coordinate measurement machines (CMM), could help to narrow down the solution space of vision systems to an easier to handle smaller number of possible system constellations or even automate system design. But although being a research topics more than two decades [3, 4], CAIP for vision systems did not make its way into widespread industrial application so far, because compared to mechanical inspection, the formation of an image is a much more complex physical process, involving object geometry, optical surface properties, light transport, optical and sensor characteristics. But while the sensor-realistic simulation of image formation is still a challenging task and subject to computer graphics research work [5], machine vision system planning can already benefit from computer aided inspection planning tools at different stages of the engineering process.

In the following we present concept and implementation of CAIP support tools for optical inspection. Beginning with annotation of CAD model files with tolerance information, followed by real time GPU-based rasterization for fast evaluation of system constellations (a constellation is set of parameters which fully describe the setup) and global illumination image synthesis for sensor-realistic image rendering. In Sect. 3.6 a versatile two-robot-setup is described, which allows to freely position camera and illumination in the hemisphere above the object of interest to create real images for a given system constellation.

2 Related Work

In their survey paper on CAD-based vision, Tarabanis et al. [6] introduced 4 categories of strategies of using computers to support the planning of optical inspection systems: 1. generate-and-test, 2. synthesis, 3. sensor simulation. 4. expert system. For a deep insight into the four groups, please refer to Tarabanis et al. [6].

Generate-and-test, e.g. the System HEAVEN [4], narrows down the solution space for a vision system design task by sampling it and testing all constellations against certain criteria, e.g. surface visibility. The synthesis approach, e.g. the work of Cowan [7], analytically calculates the positions of camera and light source for a given vision task. Expert systems, e.g. the work by Novini [8], are based on a knowledge database of already implemented vision systems or best practices and try to offer possible solutions to a new problem based on this knowledge. The work of Burla et al. [9] is based on a hybrid solution for expert system and generate-and-test by determining the visibility of surface patches using ray-tracing. Gronle et al. [10] extended this concept by adding viewpoint planning optimization. Work on the field of sensor simulation has gained attention with increasing computing power of CPU's and GPU's. Khawaja et al. [11] described an automated approach for finding the optimal positioning of camera and illumination for inspection of an assembly. Lanzetta et al. [12] used CAD-based synthetic images to optimize their error detection algorithm for a visual inspection application. Reiner [13] wrote about basic requirements to a rendering system for machine vision prototyping. He gave a general overview and presented results he achieved with simulating light sources using IES data files. He also wrote about simulating surface defects to optimize camera and illumination setup in an inspection application. Yang et al. [14] described how to use computer graphics based image synthesis to simulate the image acquisition of a surface defect.

Meister and Kondermann [15] and Retzlaff et al. [16] discussed general aspects of using synthetic images for machine vision algorithm parametrization, optimization and performance evaluation.

3 Toolset and Digital Planning Workflow

The components of our toolset and their location inside the planning and implementation workflow of the inspection system is shown in Fig. 1. Compared to the state of the art of engineering industrial vision systems, no physical sample parts are involved in the process any more. The key aspects are of the proposed workflow are: 1. Specification of inspection tasks in a machine-readable CAD format. We suggest using the ISO/PWI 10303-238 (STEP-NC) standard for this task. Being a software manufacturer independent format, STEP ensures interoperability of different CAD-systems. 2. Assigning optical surface properties to faces of the CAD model. We used a goniometer, similar to the one described by Höpe and Hauer [17], to acquire the bidirectional reflection distribution function (BRDF) of material samples cut out of a sand-casted cylinder block and fitted different mathematical reflection models to the measured data, e.g. the Cook-Torrance and Diffraction (CTD) model by Holzschuch and Pacanowski [18]. 3. Simulation of the image formation using a hybrid approach consisting of a real-time GPU rasterization rendering and a physical correct rendering using the Mitsuba renderer [19]. 4. Evaluating the results expected from image processing by calculating the lateral

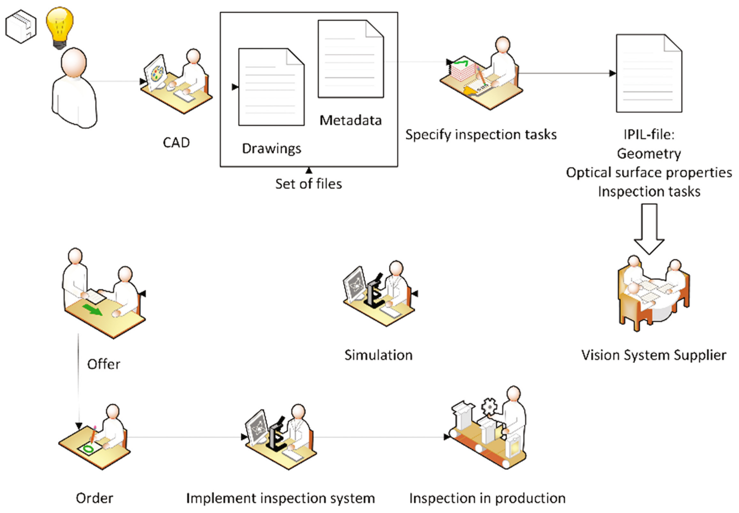


Fig. 1. Overview of the cad based planning workflow for a machine vision system [20].

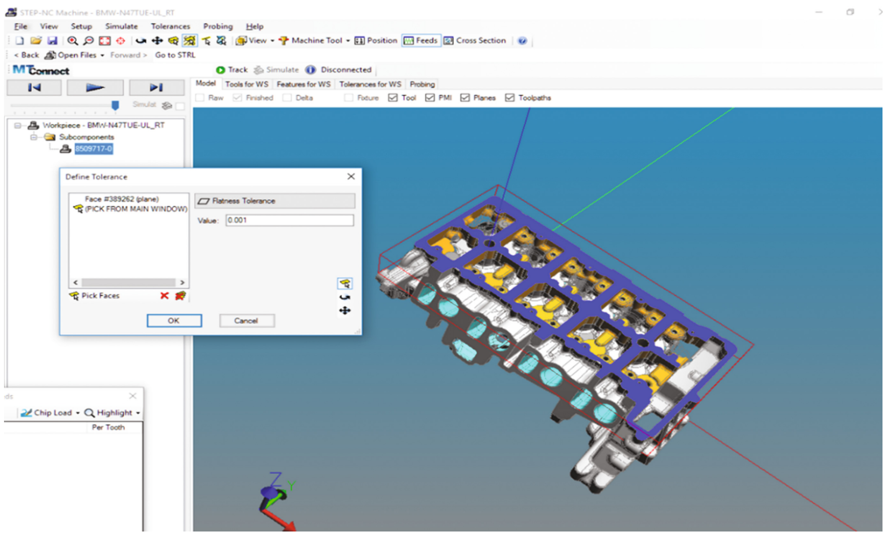
resolution and laser line detection uncertainty of the 3D reconstruction point cloud based on synthetic images. 5. Evaluating the system constellation options by using 2 industrial robots to position camera and laser.

3.1 System Overview

See Fig. 1

3.2 CAD File Annotation

The STEP AP 238 standard was used to add tolerance information and inspection tasks descriptions to faces of the CAD model as shown in Fig. 2. The geometrical dimensioning and tolerancing information was added according to the standard (ASME) Y14.5-2009 and saved away along with the geometric description (BREP) of the CAD model.


STEP-NC Machine - BMW-N47TUE-UL_RT

Workpiece - BMW-N47TUE-UL_RT
Subcomponents
8509717-0

Define Tolerance
Face #389262 (plane)
(PICK FROM MAIN WINDOW)
Flatness Tolerance
Value: 0.001

Fig. 2. Annotation of CAD files with tolerance information and inspection tasks description using STEP-NC Machine by STEP Tools (free for research use). (http://www.steptools.com/)

3.3 IPIL Data Container

In the next step, a 3D mesh was created from the CAD model, also using the STEP Tools library. We created our own application based on this library which in addition preserves the grouping of the vertices and triangles to their source faces in the CAD model. Using the open source 3D modeling tool blender[1], BRDF information was

[1] https://www.blender.org/.

added to the face groups. The 3D mesh, the material information and the inspection tasks are written into the self-developed, XML-based file format IPIL (Image processing intermediate language) [20], shown as a shorted example in the following listing:

```
<IPIL class_id="0" tracking_level="1" version="0">
  <ImageProcessingTask class_id="1" version="0">
    <GeometricToleranceToMeasure class_id="2">
      <ToleranceName>flatness_tolerance</ToleranceName>
      <ToleranceValue>0.001</ToleranceValue>
      <TolerancedShapeAspect>784</TolerancedShapeAspect>
      <Color class_id="3" tracking_level="1" version="0">
        <R>243</R>
        <G>195</G>
        <B>0</B>
      </Color>
      <FaceIdsTolerance class_id="4">
        <id>438</id>
        <id>388</id>
      </FaceIdsTolerance>
    </GeometricToleranceToMeasure>
  </ImageProcessingTask>
</IPIL>
```

IPIL file(s) can hold all information about the inspection system, including geometric configuration, camera and illumination data, “shot”-configurations describing the parameters to acquire a single frame and information about the image processing algorithm blocks and parameters. Once the digital planning is done, the vision system can be built based on the information contained in the IPIL file.

3.4 System Simulation

The real-time simulation tool which is based on GPU rasterization rendering and custom shader programs for BRDF calculations is used for interactive visualization of the expected camera sensor output as well as for a “free view” on the whole scene, allowing a fast exploration of the setup by a machine vision expert at interactive framerates (>30fps on a NVIDIA GeForce 760), shown in Fig. 3.

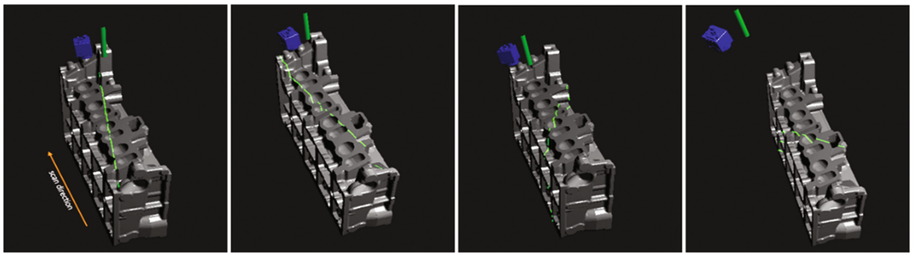


Fig. 3. "Free view" perspective in the real-time simulation tool for fast, interactive exploration of the system constellations.

3.5 System Evaluation

To evaluate the expected performance of the inspection system, different metrics are applied to the synthetic images as described by Mohammikaji et al. in [21, 22]. The lateral resolution is calculated using the real-time simulation tool which can evaluate the visibility of a surface patch from the light source(s) and the camera(s) with high speed (>30 fps). Using more GPUs in parallel would increase this number significantly. The second metric, the uncertainty, is calculated for each point on the surface by propagating the errors expected in the positioning of camera and laser and the line laser line detection error caused by object geometry and BRDF to the 3D reconstruction output [21] (Fig. 4).

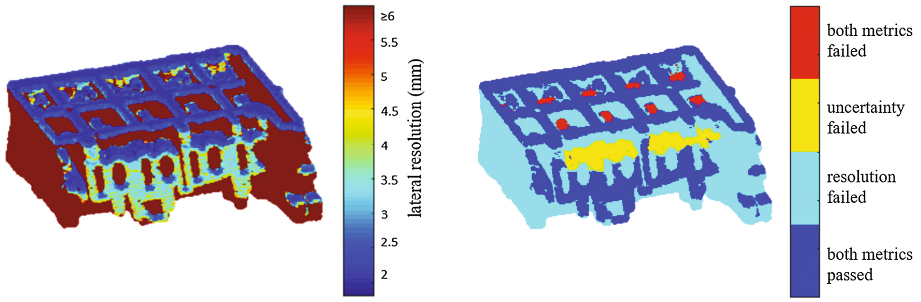


Fig. 4. Visualization of performance metrics applied to synthetic images created by the real-time simulation tool. [22] Left: Lateral resolution of the resulting 3D point cloud. Right: Visualization of object areas passing and/or failing constraints for metrics provided to the optimizer.

Figure 5 visualizes the result of the surface inference method, described by Mohammadikaji et al. [23]. The surface inference value is an indicator for the amount of information for a point the object surface which can be gained from a measurement process.

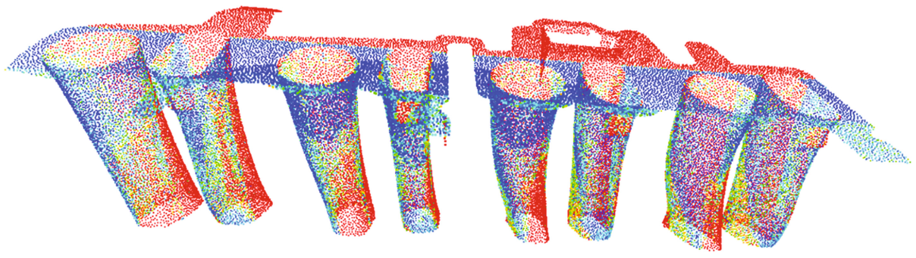


Fig. 5. Visualization of the surface inference result for the cylinder block manifolds.

3.6 2-Robot Setup for Image Capturing

In the KIT IAR/IPR-Lab a two-robot-setup, shown in Fig. 6, is used to acquire real images for evaluation of simulation results. In the setup, camera and laser are each mounted to a KUKA LWR IV lightweight robot. With this setup, camera and laser can be positioned freely in the hemisphere above the object to be inspected, only limited by the range of the robot arm and by collision avoidance. The robots are controlled via the Robot Operating System (ROS), see Fig. 7. Mission planning takes place in MATLAB, which communicates with the ROS using the MATLAB Robotics Toolbox. Mission control also triggers image acquisition, which is performed by a separate tool interfacing with the camera using GigE-Vision. Our self-developed tool "pgcapture" manages the camera parameters and saves images in raw and png format along with a JSON file containing the image and scene metadata. This metadata can be loaded in the real-time simulation tool to recreate the real system constellation in the simulation environment without manual interaction. The camera used in the setup is a FLIR (formerly PTGrey) BFLY-PGE-23S6C-C, the laser is a COHERENT SNF-501L-660FT-35-60.

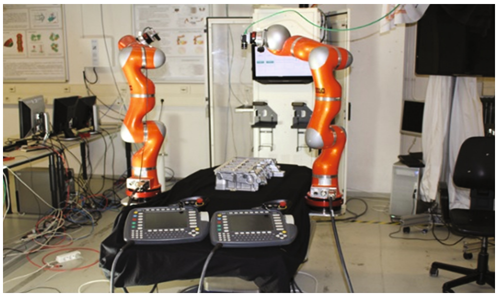


Fig. 6. Photo of the IPR lab setup showing the two-robot-setup and a sample cylinderhead object to be inspected.

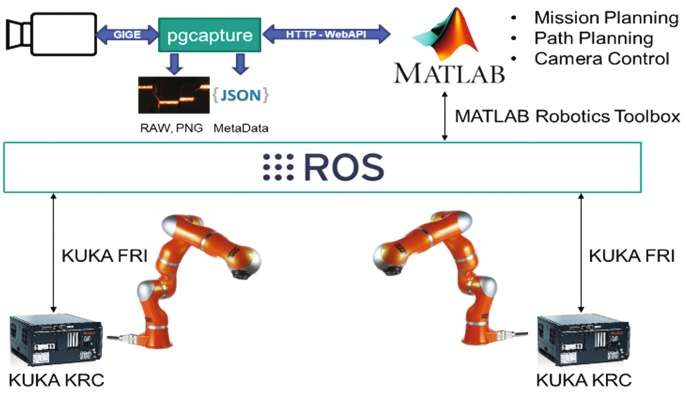


Fig. 7. System overview of the experimental setup. It consists of two KUKA LWR IV lightweight robots, controlled via the robot operating system ROS using the KUKA Fast Research Interface FRI. Images are captured using a self-developed tool "pgcapture", utilizing the API of the PointGrey camera. The tool provides a HTTP WebAPI interface to be controlled from external applications.

4 Image Acquisition Simulation and Real Images of a Laser Triangulation Setup

The experimental setup is used to acquire images for system constellations which have been selected as candidates to solve a given measurement task within the provided constraints. Figure 8 shows a collage of three images from different sources as a qualitative comparison tool for the system expert. The setup can also be used to perform scans of objects by simultaneously moving camera and laser robot in scanning direction.

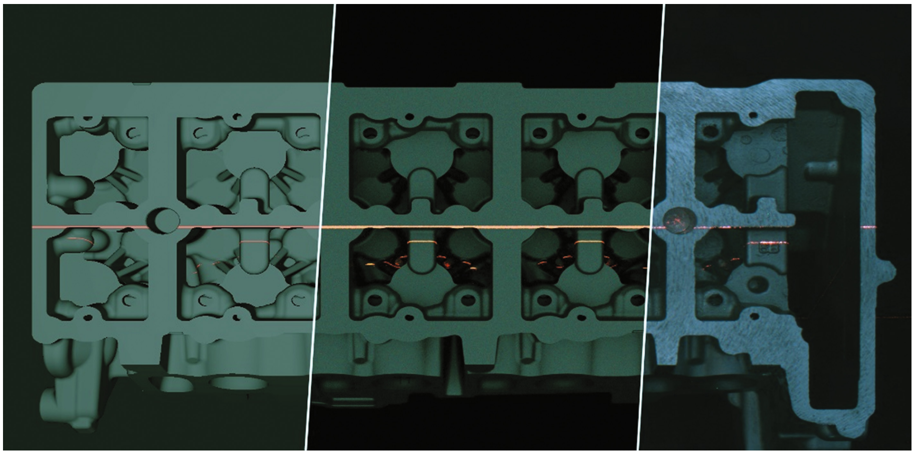


Fig. 8. A collage showing the results output of real-time simulation (left), Mitsuba rendering (center) and real camera (right) for the same system constellation (triangulation angle = 30°).

5 Summary and Discussion

In this paper, a digital workflow and a software toolset for computer aided inspection planning was presented. Using this workflow, planning of machine vision systems can take place without the dependency on physically available object samples. The digital workflow is enabled to include automatic optimization. As further steps of system planning, performance metrics to evaluate expected image processing results and an experimental setup to capture real images using two lightweight robots carrying camera and laser are described.

The workflow relies on the availability of CAD models of the objects to be inspected as well as models and parameters for camera(s) and light source(s) used. Since CAD models also contain intellectual property of the product owner, the availability of these models in industrial application is restricted or often not given. In this case, the suggested workflow is not an option. As a workaround, one could perform a 3D scan of the objects to acquire the 3D point cloud to work with, but this measurement itself introduces errors which propagate through to the simulation output. The availability of the BRDF data [20] of the objects is an important requirement. BRDF databases are very rare and their data is not acquired using comparable measurement devices and strategies. Since BRDF acquisition is a time and resource intensive process, machine vision companies could share already measured information to build up a common database of BRDF data or even start up a common measurement facility.

Acknowledgements. This research was funded by the German Research Foundation DFG as part of the project “ASP-Sim – Interaktives Computergrafik-basiertes Rapid Prototyping der Bilderfassung für die Automatische Sichtprüfung”.

References

1. Cowan, C.K., Kovesi, P.D.: Automatic sensor placement from vision task requirements. IEEE Trans. Pattern Anal. Mach. Intell. **10**(3), 407–416 (1988). doi:10.1109/34.3905
2. Al-Ahmari, A.M., Nasr, E.A., Abdulhameed, O. (eds.) Computer-Aided Inspection Planning: Theory and Practice. Taylor & Francis, CRC Press, Boca Raton (2017)
3. Tarabanis, K., Tsai, R.Y., Allen, P.K.: Overview of the MVP sensor planning system for robotic vision tasks. In: Tzafestas, S.G. (ed.) Engineering Systems with Intelligence. Springer Netherlands, Dordrecht, pp. 285–293 (1991)
4. Sakane, S., Ish, M., Kakikura, M.: Occlusion avoidance of visual sensors based on a hand-eye action simulator system: HEAVEN. Adv. Robot. **2**(2), 149–165 (1987). doi:10.1163/156855387X00138
5. Heitz, E., Hanika, J., d’Eon, E., et al.: Multiple-scattering microfacet BSDFs with the smith model. ACM Trans. Graph. **35**(4), 1–14 (2016). doi:10.1145/2897824.2925943
6. Tarabanis, K., Allen, P., Tsai, R.: A survey of sensor planning in computer vision. IEEE Trans. Robot. Autom. **11**(1), 86–104 (1995). doi:10.1109/70.345940
7. Cowan, C.K.: Model-based synthesis of sensor location. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 900–905 (1988)
8. Novini, A.R.: The lighting and optics expert system for machine vision. In: Proceedings of 29th Congress of ISPRS (1988)

9. Burla, A., Haist, T., Lyda, W., et al. An assistance system for the selection of sensors in multi-scale measurement systems. In: Furlong, C., Gorecki, C., Novak, E.L. (eds.) SPIE Optical Engineering + Applications. SPIE, 77910I (2010)
10. Gronle, M., Osten, W.: View and sensor planning for multi-sensor surface inspection. Surf. Topogr.: Metrol. Prop. **4**(2), 024009 (2016). doi:10.1088/2051-672X/4/2/024009
11. Khawaja, K., Maciejewski, A., Tretter, D., et al. Camera and light placement for automated assembly inspection. In: Proceedings of the 1996 IEEE International Conference on Robotics and Automation, vol. 4, pp. 3246–3252 (1996)
12. Lanzetta, M., Santochi, M., Tantussi, G.: Computer-aided visual inspection in assembly. CIRP Ann. Manuf. Technol. **48**(1), 13–16 (1999). doi:10.1016/S0007-8506(07)63121-7
13. Reiner, J.: Rendering for machine vision prototyping. Opt. Des. Eng. III **7100**(1), 710009 (2008)
14. Yang, H., Haist, T., Gronle, M., et al.: Realistic simulation of camera images of local surface defects in the context of multi-sensor inspection systems. In: Lehmann, P., Osten, W., Albertazzi Gonçalves, A. (eds.) SPIE Optical Metrology. SPIE, pp. 9525221–9525226 (2015)
15. Meister, S., Kondermann, D.: Real versus realistically rendered scenes for optical flow evaluation. In: 2011 14th ITG Conference on Electronic Media Technology, pp. 1–6 (2011)
16. Retzlaff, M.-G., Stabenow. J., Dachsbacher. C.: Synthetic image acquisition and procedural modeling for automated optical inspection (AOI) systems. In: Puente León, F. (ed.) Forum Bildverarbeitung 2014, pp. 47–59. KIT Scientific Publishing, Karlsruhe (2014)
17. Höpe, A., Hauer, K.-O.: Three-dimensional appearance characterization of diffuse standard reflection materials. Metrologia **47**(3), 295 (2010)
18. Holzschuch, N., Pacanowski, R.: A physically-based reflectance model combining reflection and diffraction. Research Report (2016)
19. Jakob, W.: Mitsuba 0.5.0 Documentation (2014)
20. Irgenfried, S., Wörn, H., Bergmann, S., et al.: CAD-basierter workflow für den semi-automatischen Entwurf optischer Inspektionssysteme. at - Automatisierungstechnik **65**(6), 426–439 (2017). doi:10.1515/auto-2017-0044
21. Mohammadikaji, M., Bergmann, S., Irgenfried, S., et al.: A framework for uncertainty propagation in 3D shape measurement using laser triangulation. In: 2016 IEEE International Instrumentation and Measurement Technology Conference Proceedings, pp. 1–6 (2016)
22. Mohammadikaji, M., Bergmann, S., Irgenfried, S., et al.: Performance characterization in automated optical inspection using CAD models and graphical simulations. In: Zimmermann, S. (ed.) XXX. Messtechnisches Symposium. De Gruyter, Berlin, Boston (2016)
23. Mohammadikaji, M., Bergmann, S., Irgenfried, S., et al. Probabilistic surface inference for industrial inspection planning. In: 2017 IEEE Winter Conference on Applications of Computer Vision, pp. 1008–1016. IEEE (2017)

The Impact of Web Pages' Load Time on the Conversion Rate of an E-Commerce Platform

Wiktor Stadnik and Ziemowit Nowak(✉)

Wrocław University of Science and Technology,
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland
ziemowit.nowak@pwr.edu.pl

Abstract. The conversion rate is a key determinant of effectiveness of an e-commerce platform. The impact of web pages' load time on conversion rate of an e-commerce platform was studied. During the experiment, the behavior of real store's users on the Magento platform was monitored. The popular Google Analytics reporting tool was used to collect data. The level of acceptance of page load times has been determined by the Apdex index. The results of the research showed that average page loading times have a direct impact on the e-commerce conversion rate and customer satisfaction. It was found that the use of Apdex index in different countries requires a recalculation of limits of tolerance and satisfaction.

Keywords: E-commerce conversion rate · Apdex · Google analytics

1 Introduction

The performance analysis of an e-commerce platform is a key element in its design, construction and management process. The level of customer satisfaction with system performance can have a direct impact on the level of conversion achieved in an online store. Starting up an inefficient service will result in sustaining financial losses.

To measure the effectiveness of online stores E-commerce Conversion Rate [1] is used. It is the percentage of visits which resulted in an e-commerce transaction. It determines the effectiveness of a site's marketing and design: whether its advertising attracts the most likely buyers and whether the site design makes it easier for them to shop.

The authors of this paper aimed at analyzing the performance of an online store running on the Magento platform [2] using the Google Analytics tool [3]. The tool is used to collect statistics from websites. It is capable of capturing, analyzing, and reporting the performance of a site from the perspective of the end user. Reports thus collected also help to identify critical areas of the site.

With the defined purpose in mind, the authors hypothesized that the conversion rate of an online store is dependent on the acceptability of the loading time of its pages, which may be expressed by the Apdex (Application Performance Index) [4].

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 38th International Conference on Information Systems Architecture and Technology – ISAT 2017*, Advances in Intelligent Systems and Computing 655, DOI 10.1007/978-3-319-67220-5_31

2 Related Work

In 2009, Hasan et al. investigated whether advanced web metrics, calculated using Google Analytics software, could be used to evaluate the overall usability of e-commerce sites, and also to identify potential usability problem areas. They showed that this metrics cannot do is provide in-depth detail about specific problems that might be present on a web page [5].

In 2010, Borzemski and Suchacka raised the issue of quality of Web service (QoWS) in e-commerce with the focus on request admission control and scheduling from the profit perspective of the owner of an e-business site. They presented a new approach to B2C service differentiation using RFM (Recency, Frequency, Monetary Value) analysis. They discussed a simulation model of the B2C Web server system and some results of the system performance for their approach [6].

In 2014, Poggi et al. to help set performance objectives for maximizing user satisfaction and sales, while minimizing the number of servers and their cost, they presented a methodology to determine how user sales are affected as response time increases. They also presented the evaluation of high response time on users for popular applications found on the Web [7].

In 2015, Wang and Wang suggested that user satisfaction be a key measure of computer system success. In Internet of Things, systems aim to satisfy users by assigning a suitable process sequence to massive services under users' dynamic influence. The authors proposed a dynamic priority assignment algorithm through service data distribution forecasting and user satisfaction based regulation. They calculated user satisfaction based on Apdex [8].

In 2016, McDowell et al. examined empirical associations between website features and online conversion rates through regression analysis. Results indicated that certain website design features do explain a sizeable portion of the variance converting e-commerce visitors to purchasers [9].

In 2017, Stępniak and Nowak analyzed the effectiveness of methods accelerating the process of loading applications of SPA type, including the mechanisms offered by the HTTP/2 protocol. The results indicate that the most effective technique accelerating the process of loading an SPA application is the minification of JavaScript code, which significantly reduces the size of transferred resources. The removal of unnecessary CSS rules, despite the small difference in reducing CSS files, undoubtedly has a positive impact on accelerating an SPA application [10].

3 The Measure of Application Performance

Achieving high performance of a web application is a critical issue. Apdex can be used to determine it. It is an open index developed by an alliance of companies that have defined a method standardizing the concept of system performance. It involves the processing of multiple measurements into a single number from 0 to 1 (0 = no satisfied users, 1 = all users are satisfied). The advantage of using Apdex is that it can be used for any source of end-user performance measurement. If one is in possession of a measurement tool that collects millions of results on page loading times, Apdex can help to standardize them. The measure allows one to translate a lot of response times, assessed at the user level into a single number determining the performance of a web system. Application response time is defined as the time measured from the time the user interacts with the system until the system completes the request (the user is allowed to proceed to another process) [4].

The index is based on the three user satisfaction zones:

Satisfaction. The user is fully efficient. Activities undertaken on the site are not hampered by application response time. System response times are fast enough to satisfy the user who is able to concentrate fully on his or the work with minimal impact on their thinking process.

Tolerance. The user experiences sluggish system performance but tolerates it. System responses are longer than those from the satisfaction zone. The threshold is crossed at which user performance deteriorates. System responses are returned in a time that is far from ideal, but do not in themselves compromise system performance.

Frustration. Users resigning from taking further action on the site.
Assignment to the appropriate zone is based on the response time of a given application.

To evaluation a particular system with the Apdex index, the tolerance limit (F), and the satisfaction limit (T = F/4) of the total page load time must be specified. To get the Apdex index results, the loading times of a given page should be measured and assigned to the appropriate zone and then the results should be used in the formula (1).

$$Apdex = \frac{Number\ of\ satisfactions + \frac{Number\ of\ tolerances}{2}}{Number\ of\ users} \tag{1}$$

Documentation of the presented indicator shows the method of evaluating results in a scale form. Naturally each researched system should be approached individually. The ratings, along with the values of Apdex that they determine are listed below:

- $0 \div 0.5$ – unacceptable system performance,
- $0.5 \div 0.7$ – poor system performance,
- $0.7 \div 0.85$ – acceptable system performance,
- $0.85 \div 0.94$ – good system performance,
- $0.94 \div 1$ – excellent system performance.

As can be seen, the results which vary between $0 \div 0.5$ are not acceptable, so in these cases the reason for such low performance of a web system should be analyzed. Results from $0.5 \div 0.85$ range are acceptable, but not quite satisfactory. Above 0.85 the Apdex index indicates very good system performance.

4 Experiments

4.1 Introduction

Among the many data collected by Google Analytics, the average webpage load time was qualified for analyses. To measure the level of satisfaction of web site users, the Apdex index was used. In order to mark it, the authors analyzed the distributions of average load times of the webpages as they directly affect the conversion level achieved (e-commerce service performance benchmark). The analysis was based on data from the top 5 most popular countries in terms of the number of unique sessions. It helped to understand the distribution of customer satisfaction according to webpage loading times in a given country.

Segmentation. The purpose of using additional segments during the analysis of the collected data was to select a subset useful to determine the value of the Apdex index. The segments also served to define the tolerance and satisfaction limits in each study. As a result of their use, the authors were able to analyze such subsets of data that may suggest the dependency of the conversion ratio on the level of satisfaction of the site users. Separate segments were prepared for individual experiments.

Sampling. Sampling is an analytical method involving a selection of a small random subset of data from a whole set of traffic information within the site. It allows Google Analytics to calculate data for reports faster than in the case when absolutely all information is considered. The analysis of the data subset yields similar results as in the case of full set analysis, with a much shorter processing time. Due to the use of segmentation, Google Analytics returned data samples for further analysis.

4.2 Research Position

For their research purposes the authors used a genuine online store selling clothes designed for young people from around the world. The Magento platform on which the store is based has the largest market share among other available platforms. As orders come from around the world, tracking and analyzing the data collected from different countries allowed us to look at the impact of the geographic location of the end-user on the percentage of conversions in the selected country. The research was based on data collected using Google Analytics in the period from 01.11.2016 to 30.01.2017. For the time of research, the store was correlated with Google Analytics by adding JavaScript code directly to the page (right before the closing head tag). Table 1 summarizes the basic statistics of the surveyed store. A large number of unique sessions and a variety of locations made it possible to compare the webpage load times and the level of customer satisfaction effectively.

When choosing the countries that were used in the research, two principles were followed:

- the total number of unique sessions – the greater the number within a given range of dates, the more diverse were the statistical data collected,
- e-commerce conversion rate – the larger the rate, the more sessions in a given country ended in a transaction.

Table 1. Basic statistics of the surveyed online store.

Name	Description	Value
Sessions	Number of sessions in a specified date range.	808,768
Members	The number of unique users with at least one session in the selected date range.	519,702
Views	The sum of all site view for the selected range of dates (including repetitions).	2,984,278
Pages/session	Average number of subpages visited during a session.	3
Average session duration	Average length of the session.	1 min. 56 s.
Bounce rate	The percentage of visits on the page that ended with exit from the entrance side.	47.15%
E-commerce conversion rate	Percentage of all sessions ending in a transaction.	1.26%
Transactions	Total number of transactions in the selected range of dates.	10,201

Table 2 shows selected countries with two coefficients determining their choice.

Table 2. Selected countries participating in the study.

Country	Sessions	% of all sessions	E-commerce conversion rate
Germany	275,315	34.04%	1.39%
France	98,168	12.14%	1.49%
Austria	51,033	6.31%	2.17%
USA	47,695	5.90%	0.37%
Denmark	44,112	5.45%	1.83%

The dominance of users from Germany is evident in the overall number of sessions. The least promising is the USA, where the conversion factor compared to the total number of sessions is surprisingly small. Interestingly, Austria and Denmark, with a low number of sessions, achieved a conversion rate close to or over 2%.

4.3 Description of Experiments and Discussion of Results

Experiment 1: Distribution of the entire system. Based on the collected data (aggregated mean loading times for site pages), the Apdex index was calculated. The study was conducted on a sample provided by Google Analitics. Due to its large size – in this case 25,325 – the study was a reference in the other analyzes. Marking the Apdex index for the entire store allowed us to investigate whether a particular country departs from it. In order to calculate the Apdex index for the first experiment, the following steps were taken:

1. Performing the analysis of the aggregate average loading times of the site webpages.
2. Determining the tolerance limit of the Apdex index (F) based on the value of the conversion index.
3. Calculating the satisfaction boundary (T).
4. Counting the satisfied and tolerant users in the sample based on the selected Apdex limits.
5. Calculating the Apdex index and determining the level of satisfaction.

The analysis from point 1 was based on segmentation. Due to the nature of the study two segments were used:

- **All users** – the total number of BitterSweet Paris users in the selected period.
- **Made a purchase** – the total number of transactions across the store (condition: transactions per user > 0) within the specified range of dates.

The use of both segments resulted in generating a report with the indicated parameters (Table 3).

Table 3. The results of segmentation in experiment #1.

Segment	Avg. page load time [s]	Number of loading pages	Sample size for loading pages
All users	4.77	2,991,661	25,325
Made a purchase	4.14	350,295	1,971

As can be seen, the size of the sample considerably decreased in the case of people who made a purchase. Unfortunately it resulted in a decrease in the conversion achieved.

For further analysis, load times for all pages of the site (views) visited were determined on a sample of all visitors to the site and those who made the purchase. In each interval the conversion factor was determined according to the relationship:

$$\%\,conversions = \frac{Numerousness\ of\ range\ in\ Made\ a\ purchase\ segment}{Numerousness\ of\ range\ in\ All\ users\ segment} * 100\% \tag{2}$$

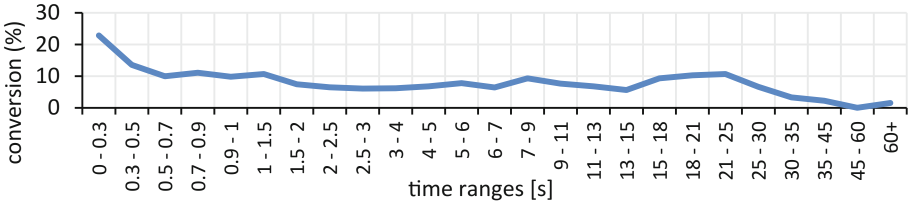


Fig. 1. Conversion rates for individual time ranges of page loading.

Figure 1 shows the determined conversion ratios within the respective time intervals. Based on this, the authors determined the moment when the conversion rate drastically decreased relative to the page load time (range of 21 ÷ 25 s of page load time). In this way, the tolerance limit (4T = 25 s) and the satisfaction limit (T = 6.25 s) of the Apdex index were established.

The next step was to sum the size of ranges in a segments *all users* to the satisfaction limit (0 ÷ 6 s) and to the tolerance limit (6 ÷ 25 s). The result of adding is presented in Table 4. The number of satisfied users is the number of satisfaction, while the number of tolerant users is the number of tolerances.

Table 4. The number of satisfied users and the number of users who tolerate loading times of the site.

Number of satisfactions	Number of tolerances
18 480	4 443

Having collected all the data, the authors marked the Apdex: $\frac{18480+\frac{4443}{2}}{25325} \approx 0,82$. The result obtained determines the researched site as acceptable in terms of its performance. The tolerance limit established by the authors proved to be a very good choice, as the result of the Apdex index did not reach a value close to 0 nor a value above 1 (ideal system performance). Approximately 9.5% (2,408 users) might have negatively preceived such a long time of page loading and resigned from the transaction.

Experiment 2: Distribution according to country. For the 5 most popular countries (Table 2) in terms of the number of unique sessions, the load time distribution of the site was analyzed. Subsequently, the Apdex index was calculated for each country (on the basis of achieved conversions). The calculation of the Apdex index for each country took place in the five steps presented in experiment 1.

The segmentation results for the USA are far from the rest of the countries. The average page load time was almost 1.5 times longer than that of the rest of the surveyed countries. It might be due to the location of the server in the center of western Europe (considerable remoteness from the USA).

Table 5 summarizes the obtained results by country.

Table 5. Summary of experiment #2 results.

Country	Avg. page load time [s]	Sample size	Tolerance limit (F)	Satisfaction limit (T)	Number of satisfied users	Number of tolerant users
Germany	4.53	8,971	21	5.25	6,840	1,986
France	5.52	3,423	13	3.25	1,732	1,509
Austria	4.96	2,083	13	3.25	1,080	970
USA	7.64	703	13	3.25	203	467
Denmark	3.86	1,115	13	3.25	789	353

The results obtained allowed us to summarize the acceptability of page load times depending on the country being measured. Table 6 shows the level of application performance according to the Apdex index obtained.

Table 6. Web application performance determined by the Apdex index.

Country	Apdex	System performance
Germany	0.87	good
France	0.73	acceptable
Austria	0.75	acceptable
USA	0.62	weak
Denmark	0.87	good

As can be seen, 4 out of 5 selected reached acceptable application performance and the US was the only country with a weak one. It might be the result of too small a sample adopted in the study. Specified tolerance limits indicate that users in Germany best tolerate long loading times. In the range 0 ÷ 21 s users from Germany make a transaction in the store. In other countries the tolerance limit was 7 s lower.

Experiment 3: Global approach to the Apdex index. The impact of average page loading times in each country and the acceptability of the Apdex index on the conversion achieved in the online store is shown here. During this experiment, data segmentation was not used because the analysis was based on the results from previous experiments.

The experiment was reduced to:

- country-specific tabular statistics, including average page load times, the Apdex index and e-commerce conversion reached,
- comparative analysis of the results collected.

Table 7 shows the statistical summary of all the countries participating in the experiment. Each country was assigned the Apdex index calculated during experiment 1, as well as additional measurements, such as page bounce rates.

Table 7. Summary of experiment #3 results.

Country	Sessions	Bounce rate	Avg. page load time [s]	E-commerce conversions rate	Apdex
Germany	275,315	45.41%	4.53	1.39%	0.87
France	98,168	43.49%	5.52	1.49%	0.73
Austria	51,033	44.36%	4.96	2.17%	0.75
USA	47,695	78.11%	7.64	0.37%	0.62
Denmark	44,112	44.86%	3.86	1.83%	0.87

While analyzing the results collected in Table 7, a certain correlation between the Apdex index and the bounce rate of the page was noticed. The smaller the measure of application performance, the greater the bounce rate of the page is. The dependency is not linear. This is well illustrated by the example of the USA which, with the Apdex index of 0.62, reached a bounce rate result of 78.11%. In addition, it was noted that the average load time relative to other countries was the highest. The same dependency can be seen with the e-commerce conversion rate. With the decrease in customer satisfaction, this factor decreases. The dependency of the e-commerce conversion factor from the Apdex index is illustrated in Fig. 2.

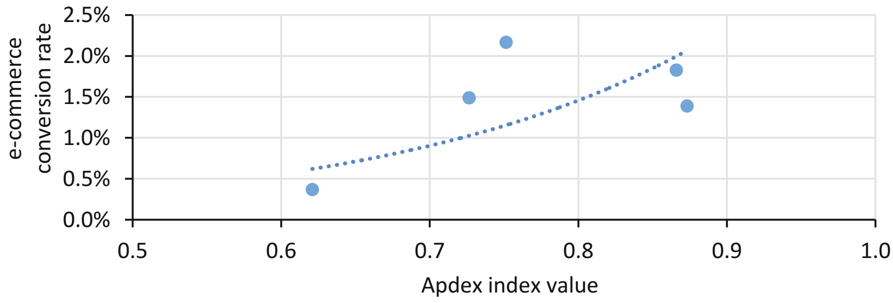


Fig. 2. The dependency of e-commerce conversion rate on the acceptability of page load times.

The trend line, marked with a dotted line, represents the expected increase in e-commerce conversion rate relative to the growing value of the Apdex index.

5 Conclusions and Directions of Further Work

The results obtained confirm the hypothesis advocated earlier by the authors. The e-commerce conversion rate depends on the satisfaction and acceptability of the page load times as measured by the Apdex index.

The small sample size imposed by Google Analytics might have affected the results of the experiments.

The average loading times of webpages in specific countries may have been dependent on the location of the server. For European countries the average was 4.72 s and for the USA it was 7.64 s. Selecting a country located on another continent allowed the authors to identify the tolerance range of the site loading times. Of course, the selected countries were to be among those with the highest number of unique visits. Running a mirror server in the US would greatly improve the site's performance.

The time interval mappings recorded by Google Analytics allow for a comprehensive analysis of the time periods of all stages of a page's loading. In the conducted studies, the use of average page load time only was crucial due to the direct impact on the store conversion.

Web-based performance evaluating requires an individual approach. Determining the universal limits of the Apdex index is impossible due to the variety of the systems evaluated. Therefore, it is important that the limits of tolerance and satisfaction are based on real users' data taken from reports offered by e.g. Google Analytics.

Google Analytics offers a number of additional dimensions such as the browser or the device used to display the indicated page. Singling some of these out would help in a broader analysis of the performance of web systems. Extension of the research by the time intervals collected but not used during the analysis would result in the examination of the impact of individual loading steps on the generally perceived performance of the web system.

By default, Google Analytics generates reports based on a specific group of users only. In the analysis of services with a small sample of visits it is worth improving the quality of the data provided by increasing the sample rate.

A customized tool that would provide more personalized reports on directly online stores based on the Magento platform could be developed. The analysis of the results returned by such a tool would make it possible to determine the reliability of the data collected with Google Analytics.

References

1. Rabhan, B.: Convert Every Click: Make More Money Online with Holistic Conversion Rate Optimization. Wiley, Hoboken (2013)
2. Khliupko, V.: Magento 1 DIY. Apress, New York (2016)
3. Clifton, B.: Advanced Web Metrics with Google Analytics, 3rd edn. Wiley, Indianapolis (2012)
4. Gunther, N.: The Apdex Index Revealed. MeasureIT 7.02. Computer Measurement Group (2009)
5. Hasan, L., Morris, A., Probets, S.: Using Google analytics to evaluate the usability of e-commerce sites. In: Kurosu, M. (ed.) Human Centered Design 2009, LNCS, vol. 5619. Springer, Heidelberg (2009)
6. Borzemski, L., Suchacka, G.: Business-oriented admission control and request scheduling for e-commerce websites. Cybern. Syst. **41**(8), 592–609 (2010). SI
7. Poggi, N., Carrera, D., Gavaldà, R., Ayguadé, E., Torres, J.: A methodology for the evaluation of high response time on E-commerce users and sales. Inf. Syst. Front. **16**(5), 867–885 (2014)
8. Wang, G., Wang, Y.: User satisfaction based dynamic priority assignment algorithm for internet of things. In: 17th International Conference on High Performance Computing and Communications, pp. 880–883. IEEE, New York (2015)
9. McDowell, W., Wilson, R., Kile, C.: An examination of retail website design and conversion rate. J. Bus. Res. **69**(11), 4837–4842 (2016)
10. Stępniak, W., Nowak, Z.: Performance analysis of SPA web systems. In: Borzemski, L., Grzech, A., Świątek, J., Wilimowska, Z. (eds.) Information Systems Architecture and Technology 2016. Advances in Intelligent Systems and Computing, vol. 521. Springer, Cham (2017)

Nearest Neighborhood Determination Methods with Regard to Mobile Location-Based Application

Mariusz Fraś(✉), Piotr Frącek, and Krzysztof Waśko

Wrocław University of Science and Technology, Wrocław, Poland
{mariusz.fras,krzysztof.wasko}@pwr.edu.pl,
192201@student.pwr.edu.pl

Abstract. The paper concerns nearest neighborhood determination issues. The main goal is to investigate and compare characteristics of popular solutions with regard to applying them to location-based services for mobile devices. Mainly the impact of number of objects and object's mobility on performance is considered. Also the accuracy dependent on search range was tested. The paper takes into account three groups of approaches: brute-force based solutions, spatial tree structures, and solutions that use Geohash for location encoding. Described simulation experiments was performed for eight popular algorithms and one own approach. The results show very diverse properties of all algorithms and its usefulness for considered area of application.

Keywords: Nearest neighborhood · Spatial searching · Location-based services

1 Introduction

The development of mobile technology makes mobile-based services available to users increasingly location-based. This means that the data provided by the applications largely depends on the location. According to research conducted by Latitude, 61% of users have a better view of the ad when the location is taken seriously. Various studies have reported that the value of the location-based application market will increase from 10.3 billion in 2014 to 34.8 billion in 2020 [1]. So the trend is visible and it should be expected to keep it in the coming years.

Nowadays, the solutions based on the user's location are developed by large corporations as well as small companies. The importance of research can also provide existing commercial applications for finding objects at a given distance, e.g.: Facebook Messenger – the app itself displays information about friends nearby, Tinder – a dating application that lets you find users at a given distance, Google Maps – it allows you to search for nearby places (e.g. pharmacies, petrol stations, etc.).

Geographic coordinates, due to their two-dimensional nature, are described using spatial indexes. Standard indexing methods that are used in one-dimensional databases do not support spatial data-like operations such as finding elements in a given region [2]. This operation is crucial for finding points at a given distance.

Examples of such an operation are: to find all the ambulances in the vicinity of the motorway, to send a message to students currently arriving on campus, or to find all night pharmacies within a given radius. It should be noted that the given examples differs from searching point of view. The set of pharmacies (buildings) is much less dynamic than the set of moving ambulances or people. The set of people will be usually much more numerous than a set of ambulances. This means that the conditions that the spatial index should fulfill depend on the application.

Many commonly used spatial indexing structures (such as the R-tree) were created long before the emergence of mobile devices with geo-targeting capabilities. They changed the requirements of the available solutions. Technological progress could also have led to the disappearance of certain performance problems, and the growing market for mobile devices, the emergence of other, like the search for objects at a given distance.

The main aim of this study is to analyze and compare the solutions available to determine the nearest neighborhood (i.e. to find all objects (points) at a given distance from the given center point) in terms of:

1. Performance depending on the size of the dataset (number of objects) and on data set dynamics (the frequency of change of object position).
2. Accuracy depending on search range (distance from the center point).

This work is designed to identify the strengths and weaknesses of particular algorithms, and to identify significantly better solutions for specific cases. It is first step to help all who are implementing solutions based on the location of mobile devices.

The paper takes into account three groups of approaches: brute-force based solutions, spatial tree structures, and solutions that use Geohash location encoding. Experiments was performed for eight popular algorithms and one own approach - modification of Geohash-based algorithm called B+ Tree Integer Geohash.

The research involved simulating the operation of the algorithms in the real world. The tests were conducted in a specific environment (hardware configuration), which could have affected the results. The results obtained should therefore be treated only as a comparison of the described solutions and their disadvantages and advantages and possible applications.

2 Short Characteristic of Considered Algorithms

For computational experiments nine algorithms from three groups were taken into consideration (Table 1). The fundamental differences between different approaches are the complexity (or complicatedness) of basic operations necessary for the process of finding nearest moving objects, i.e. search (for given center point find a set of objects within given distance), update (change data of given object) and insert (insert new object – necessary to update an object in some methods), and that non complete search methods for search nearest objects query returns so called candidate objects, i.e. narrows set of all objects to a set of objects with some distance accuracy.

Table 1. List of tested algorithms with structure of data organization and codenames.

Type	Structure	Code name
Complete search	Array	BruteWGS84
Complete search	Array	BruteHaversine
Complete search	Array	BruteEuclidean
Classic spatial index	R-tree	RTree
Classic spatial index	Quad-tree	QuadTree
Geohash	Associative array	GeohashMap32
Geohash	Associative array	GeohashMap2
Geohash	Ternary search tree	GeohashTree
Geohash	B+ tree	GeohashBPlus

2.1 Basic Popular Algorithms

Complete search based approaches that calculate distances between the reference point and all other points are cost-expensive but basically most accurate. Some algorithms apply simplifications that decreases computational cost but also decreases accuracy. The tested algorithms from this group is Vincenty's algorithm based on reference system WGS84, Haversine formula, and Euclidean distance algorithm.

The Vincenty's formulae based algorithm is iterative method of calculations [3]. It assumes approximation of Earth by an oblate ellipsoid, a slightly flattened sphere. In calculations the World Geodetic System 84 (WGS 84), the approved standard in geodesy, satellite navigation and cartography is used. The number of iterations depends on considered points and it ranges from a few to several thousands. It is assumed that a sufficient accuracy is an error 0.000005555556°, or about 0.5 mm, what is much less then GPS accuracy. That's why it is assumed in the work as the reference algorithm for evaluation of accuracy of the rest of the algorithms.

The Haversine formula-based algorithm assumes the globe as an ideal sphere. It calculates the orthodrome, the shortest possible distance between two points on the sphere. It has a much lower computational cost (it is not iterative approach). The maximum error of the orthodrome is 0.5% [4].

When calculating the distance on the globe we mean the distance on its surface, which is not a distance in Euclidean sense. However, it is generally accepted that the curvature of the earth can be omitted at short distances. In algorithm computing Euclidean distance it is assumed the simplest form of calculations [5] where the longitude distance and latitude distance between point *i* and center point are calculated (in meters) using formula (1).

$$\begin{cases} \Delta lat_i = (lat_0 - lat_i) \cdot k \\ \Delta lon_i = (lon_0 - lon_i) \cdot k \cdot \cos(lat_0) \end{cases} \tag{1}$$

where: lat_i, lon_i – are latitude and longitude of point *i* (index 0 for center point), and $k = 110250$.

Because for search operation examining the distance requires iteration over all objects, the structure used in complete search methods in this work is an array that provides easy access to data. Update operation requires knowledge of the location of the point in the array (index) or the need to search it. In the research array filtering was used because due to asynchronous simulations it should be taken into account that the indexes may change. The insert operation is accomplished by exchanging references to the objects.

Next group are classic tree structure based algorithms used in geodesy. The basic R-tree and Quad-tree algorithms are tested. R-tree is a structure that allows to store not only points but also whole areas. The algorithm invented by Guttman [2] is widely used in database systems and advanced geodetic systems. In the R-tree all leaves are on the same level (it is balanced tree). Each leaf consists of a stored object and a minimum region (which is a rectangle) covering that object. Each non-leaf vertex is made up of a descendant set and a minimum region covering each item in the set.

For R-tree we search all points in the specified region. The circle (centered at the center point with radius of the search range) should be transformed into a minimum overlap region. Then algorithm search which descendant regions overlap it, and repeats procedure for them. As a result we receive also objects outside the search radius. Update is a three-fold operation: finding a point, removing it, inserting a point with an updated position. Insert operation is a somewhat more complex operation.

The initial version of Quad-tree created by Raphael Finkel and J.L. Bentley [6] has many variants today and are used not only in geodetic applications. In this work the Polygon Map Random (PMR) Quad-tree is studied, which resembles the R-tree but exhibits better performance for dynamic data [7]. The basic feature of the studied structure is the recurrent partition of space into four equal quarters. Each node is described by the x, y coordinates, and height and width. The search, update and insert operations are similar to R-tree approach, and due to rectangle regions, as e result of search we receive also objects outside the search radius.

The last group are popular in recent years algorithms based on Geohash geocoding system. Three solutions that uses associative array and ternary tree, and own solution B+ tree structure based Integer Geohash algorithm are considered.

The Geohash-based algorithm divides the land map into two equal parts. One is marked with zero and the other one. It then proceeds similarly to each of the parts until the desired precision is obtained, which is the cell size of the mesh obtained by the division. Geohash is not a data structure. This is a solution that describes the coding (and decoding) method of coordinates. Based on the coordinates of a point it is defined his Geohash which is a long series of zeros and ones. To reduce the number of characters, it is converted to a character string of base 32. The number of characters (code length) indicates its accuracy. Points that are close to each other have a common prefix that is greater, the smaller the distance between points. This is important when searching for a neighborhood. To find set of points within given distance you must specify prefixes (with proper precision) that may have the codes of the searched points, and then extract from the used data structure the points that contain any of the specified prefixes.

The simplest implementation of the method for finding points based on Geohash similarity is the use of an associative array where points are stored in the following form: < Geohash;Point > , where the Geohash code is the key. The principle of operation of 2-bit and 32-bit versions is the same. For a search operation, when using an associative array, a regular expression is created that allows you to select from the list of all keys that start with one of the desired prefixes. The found points go to the candidate points set. Update is very simple. You need to know the location of the point before the shift to calculate Geohash, which is the key in the associative array. Geohash is then recalculated and updated. Insert is reduced to calculating a key that is assembly of a Geohash code and a unique point identifier.

Searching for all text strings having a common prefix can be supported by using the data tree structure called Ternary Search Tree (TST) proposed by J.L. Bentley and R. Sedgewick [8]. The basic feature of this structure is that each node consists of a character, a value (optional) and a maximum of three descendants called "smaller", "equal" and "greater" [9]. Names indicate how to walk the tree while searching given string of characters. Insertion of characters consists in adding subsequent characters walking the tree and creating "equal" node if there is no branch for given prefix of string or creating "smaller", or "greater" node if a part of prefix exists. Search operation returns value of found "equal" node of last character (if found) and all values from "equal" branches. Update is similar to tree-based solutions. After removing node there may be a need to rebuild the structure to eliminate unnecessary vertices.

2.2 B+ Tree Integer Geohash Algorithm

In B+ Tree Integer Geohash (BTIG) algorithm we propose combine B+ tree data structure with integer form of Geohash to support searching for objects. The B+ tree [10] allows to easily find all the numbers in a given range and is used in many popular file systems (e.g. BFS, XFS, NTFS), and supported by known databases (e.g. MS SQL Server, SQLite, Oracle 8).

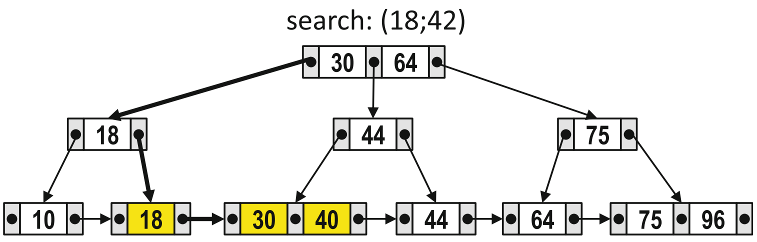


Fig. 1. B+ tree example ($M = 2$) and search(18;42) operation.

In the B+ tree (Fig. 1) the data is sorted and stored only in leaves. In addition, all the leaves are on the same level. Each node can store a minimum of $M/2$ data (M is a parameter) and maximum M data (except the root). The distinguishing feature is that each leaf has references to neighboring leaves, so that the lowest tree level resembles a bidirectional list. Finding one value (e.g. the outset of searched interval) it is very easy to find more values (bigger or smaller).

Geohash binary code can be treated not as a string, but as a number that can be converted to a decimal. This form takes up less space and the operations performed on it are much simpler. A similar form is used by the Redis database. This way, searching for points based on a prefix can be easily replaced by searching for numbers within a given range. By establishing a constant Geohash code precision in binary form and knowing the length of the prefix (it defines search range), it is easy to generate the smallest and largest possible number with the given prefix. For given prefix length for search, the boundary values are obtained by setting the rest of bits of Geohash to zero and to one and converting Geohash to decimal (see example on Fig. 2 for binary code length equal 52 and center point Geohash(10) = 4237676461477541 (Geohash (2) = 1111000011100010010011110000001010001100111010100101), and prefix length for search equal 18).

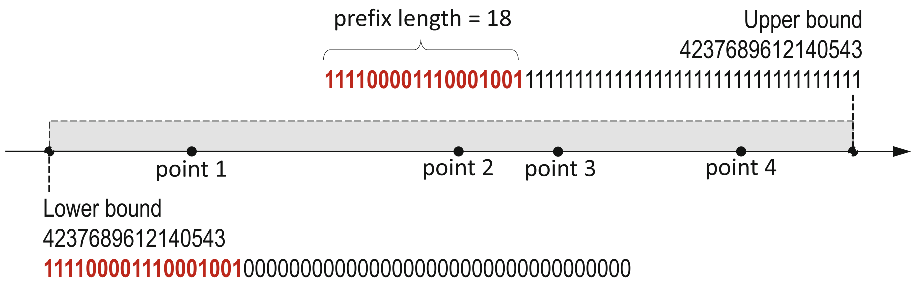


Fig. 2. Example of search range for Geohash(10) = 4237676461477541 and prefix length 18.

The search consists in determining the bounds, finding a leaf with value equal lower bound (or greater if no node with the same value is found), and finding all points with values less or equal the upper limit. Insert operation may require to rebuild part of the tree structure. Update is similar to tree-based solutions.

3 Performance and Accuracy Tests

3.1 Research Methodology

The tested algorithms (except for the complete search ones) allow to found candidates only, i.e. allow to significantly reduce the set of points, from which subsequent selection of points satisfying the neighborhood search criteria should be made. In this paper, the accuracy of the neighborhood search algorithm is defined as the ratio of the number of points satisfying the search conditions to the number of candidates. The performance of the algorithm is defined as the number of performed neighborhood search operations in a fixed time unit (same for all algorithms).

The research involved simulating the operation of the algorithms. All algorithms were implemented in JavaScript and tested on a virtual OVH VPS CLOUD RAM 2 server (two cores, 2.4 GHz, 12 GB RAM).

Each of the algorithms was tested 50 times for each possible interval configuration between update operations (Table 2) and number of points (Table 3), which resulted in 36,000 independent tests, each of which lasted 10000 ms.

Table 2. Parameters of interval distribution between update operations.

Parameter	1	2	3	4	5	6	7	8	9	10
Expected value [ms]	0,05	0,1	0,2	0,5	1	2	3	5	10	15
Standard deviation [ms]	0,005	0,01	0,02	0,05	0,1	0,2	0,3	0,5	1	1,5

Table 3. Number of tested points

	1	2	3	4	5	6	7	8
Number of points	10000	100000	500000	1000000	1500000	2000000	2500000	3000000

The work assumes initial even distribution of the objects (points) across the area. During each simulation (each lasting 10000 ms) asynchronous queries to the server that generated update (change of point position) and search operations were performed. Operations of point localization were separated by random interval value with normal distribution. Each of the algorithms was tested for all interval distribution parameters between updates (Table 2). Values were selected experimentally to simulate both large and small server load. During each update, the location of a single random point was changed by a maximum of 0.001° latitude and 0.001° longitude, or 112 meters on the equator. This action was intended to model the movement of indexed in the database points corresponding to, for example, the movement of people, vehicles, drones, etc.

Searches, as well as updates, were separated by an interval of normal distribution with an expected value of 24 ms and a standard deviation of 10 ms. The values were chosen experimentally so as to heavily load the server but letting it work. The search input, i.e. the center point and the search range (from 30 m to 100 km), were also randomly selected to model searches on both small and large distances.

3.2 Simulation Results

The performance was tested against the frequency of object position changes (object dynamics) and number of objects (as in Tables 2 and 3). The Figs. 3, 4, and 5 show the performance versus update interval (the expected value of interval distribution) for number of objects equal 100000, 1500000, and 3000000 respectively. As expected, the complete search approaches are significantly less efficient even for relatively small number of objects (for 10000 objects the difference is lower).

Also Geohash-based methods with associative array have poor performance. As number of objects increases also RTree and QuadTree algorithms become distinctly less effective than most efficient GeohashTree and proposed GeohashBPlus algorithms. Only for very large number of objects (3 milion in Fig. 5) GeohashBPlus works a little less efficiently than GeohashTree. On the other hand GeohashBPlus seem to have better stability of accuracy than slightly more efficient GeohashTree.

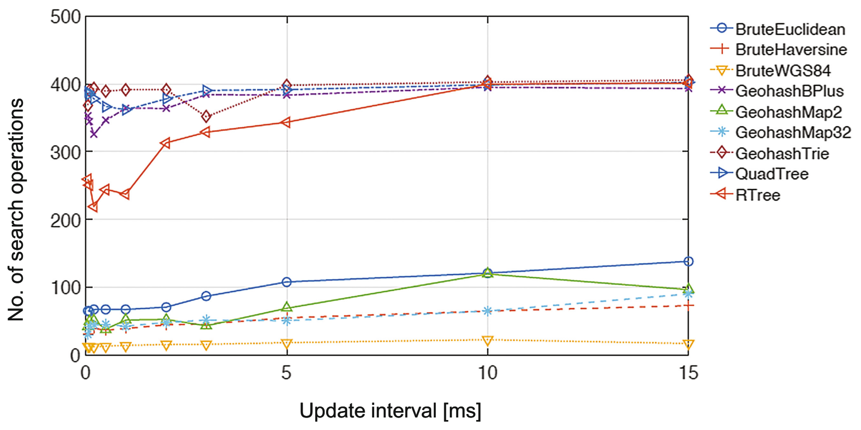


Fig. 3. Performance versus the frequency of object position changes for 100000 objects.

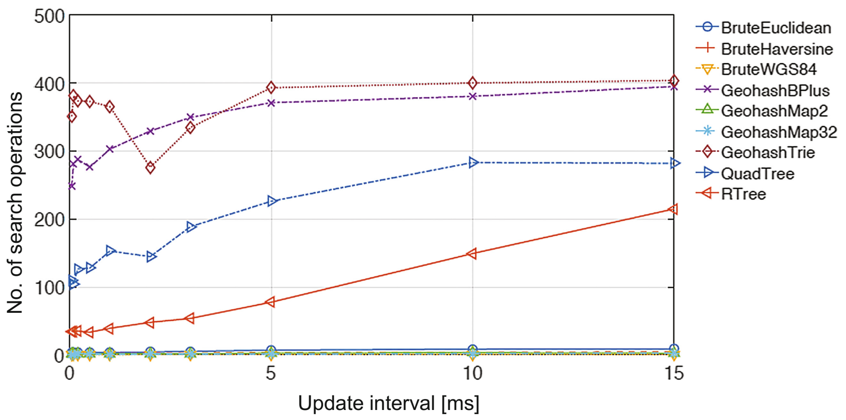


Fig. 4. Performance versus the frequency of object position changes for 1500000 objects.

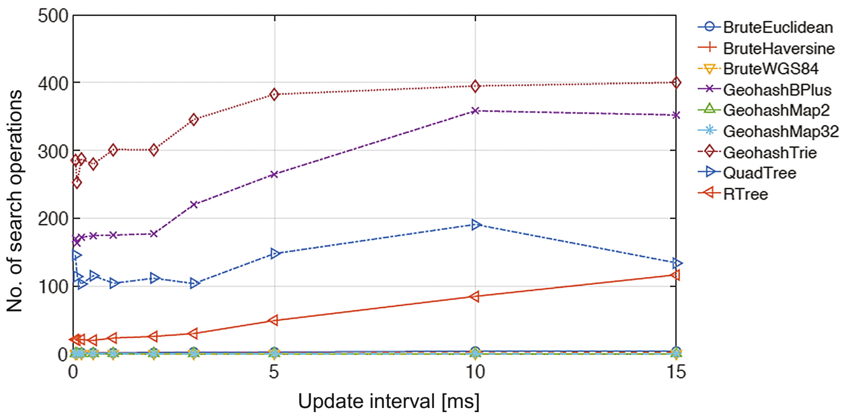


Fig. 5. Performance versus the frequency of object position changes for 3000000 objects.

The Figs. 6 and 7 show measured accuracy versus the search range (Fig. 6 for distance up to 10 km and Fig. 7 for distance up to 100 km).

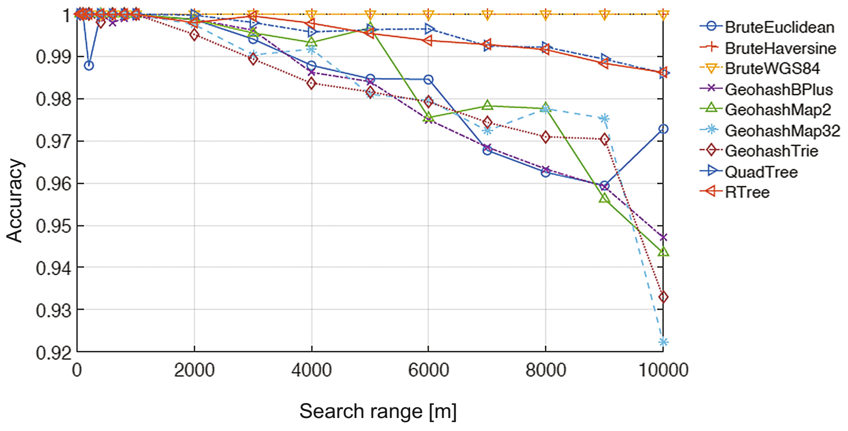


Fig. 6. The accuracy versus the search range (distance 0 ÷ 10 km).

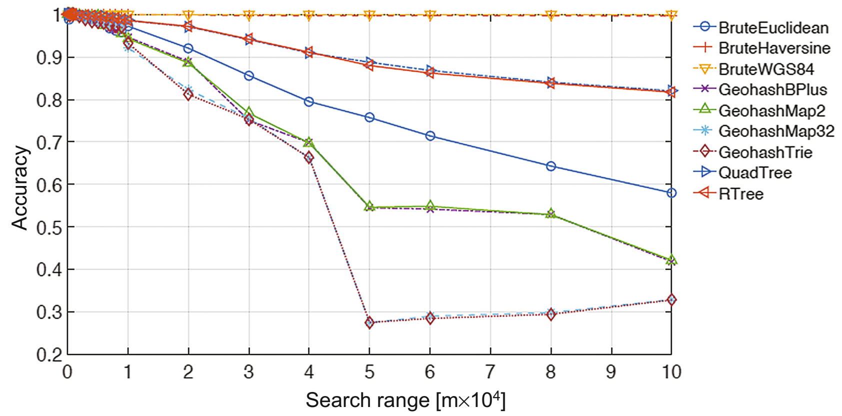


Fig. 7. The accuracy versus the search range (distance 0 ÷ 100 km).

For small distances (Fig. 6) differences in accuracy are clear but all algorithms work well. The accuracy of all solutions is over 92%. This changes for greater distances. The Geohash-based algorithms works worse, but draws attention significant decrease of accuracy of GeohashTree for ranges over 40 km. The GeohashBPlus is also less accurate for greater distances but decrease is not so rapid. It is interesting that more efficient BruteHaversine is almost as accurate as reference BruteWGS84, and that BruteEuclidean is worse than classic tree structure based solutions.

4 Final Remarks

In the work selected algorithms were tested for performance and accuracy for different number of objects and intervals between updates of its location. Additionally, the impact of the maximum distance for search was investigated. The performed experiments show that the performance of algorithms seems to be more dependent on the number of stored points (objects) than on objects dynamics. After a certain performance limit has been reached, the frequency of the location changes has a significant effect on the accuracy of the algorithms. The most resistant are the Geohash-based algorithms, but with a tree-based data structure.

The results show in what conditions what algorithms should be applied. The Haversine formula based algorithm is almost as accurate as the one based on Vincenty's formula, but with better performance. Thanks to this, it can be used as a substitute, but only in solutions always requiring very high precision. When we expect heavy load (number of objects with frequent updates) the tree structure based solutions are the best. For heavy load and not large distances Geohash-based methods with use of tree-based data structures seems to be good choice. However additional solutions that improve its accuracy should be applied.

References

1. Malm, A.: Mobile Location-Based Services, 9th edn. Berg Insight AB (2015)
2. Guttman, A.: R-trees: a dynamic index structure for spatial searching. In: Proceedings of ACM Management of Data (SIGMOD), pp. 47–57 (1984)
3. Vincenty, T.: Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations (1975)
4. Great Britain: Ministry of Defence, Admiralty Manual of Navigation, vol. 1. Stationery Office, London (1997)
5. Salonen, J.: Geographic Distance Can Be Simple and Fast (2014). http://jonisalonen.com/2014/computing-distance-between-coordinates-can-be-simple-and-fast accessed 31 Jul 2017
6. Finkel, R.A., Bentley, J.L.: Quad trees a data structure for retrieval on composite keys. Acta Informatica **4**(1), 1–9 (1974). Springer, New York
7. Hjaltason, G.R., Samet, H.: Speeding up construction of PMR quadtree-based spatial indexes. VLDB J. **11**(2), 109–137 (2002)
8. Bentley, J.L., Sedgewick, R.: Fast algorithms for sorting and searching strings, SODA 1997. In: Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, New Orleans, pp. 360–369 (1997)
9. Sedgewick, R., Wayne, K.: Algorithms, 4th edn. Addison-Wesley, Boston (2011)
10. Comer, D.: The Ubiquitous B-Tree. ACM Comput. Surv. **11**(2), 121–137 (1979)

Author Index