Piotr Gaj
Michał Sawicki
Grażyna Suchacka
Andrzej Kwiecień (Eds.)

Communications in Computer and Information Science 860

Computer Networks

25th International Conference, CN 2018
Gliwice, Poland, June 19–22, 2018
Proceedings

Springer

Communications in Computer and Information Science 860

Commenced Publication in 2007
Founding and Former Series Editors:
Alfredo Cuzzocrea, Xiaoyong Du, Orhun Kara, Ting Liu, Dominik Ślęzak, and Xiaokang Yang

Editorial Board

More information about this series at http://www.springer.com/series/7899

Piotr Gaj · Michał Sawicki
Grażyna Suchacka · Andrzej Kwiecień (Eds.)

Computer Networks

25th International Conference, CN 2018
Gliwice, Poland, June 19–22, 2018
Proceedings

Springer

Editors
Piotr Gaj
Institute of Informatics
Silesian University of Technology
Gliwice
Poland

Michał Sawicki
Silesian University of Technology
Gliwice
Poland

Grażyna Suchacka
University of Opole
Opole
Poland

Andrzej Kwiecień
Silesian University of Technology
Gliwice
Poland

ISSN 1865-0929 ISSN 1865-0937 (electronic)
Communications in Computer and Information Science
ISBN 978-3-319-92458-8 ISBN 978-3-319-92459-5 (eBook)
https://doi.org/10.1007/978-3-319-92459-5

Library of Congress Control Number: 2018944399

Printed on acid-free paper

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

Preface

In 1993, our dear colleague, prof. Andrzej Grzywak, was thinking about how to integrate the Polish academic community involved in computer networks research and development. On the one hand, it was a time of great progress in network usage and application, both in local and wide area network types. On the other hand, at this time the public Internet was just beginning; it was just started as an open and global network and its range and availability were incomparable to those of today. Therefore, the common access to knowledge dissemination and lookups was limited. Moreover, the number of scientific journals related to computer networks domains and their quality were not as high as today and access to them was also imperfect. Thus, there was a need to create an opportunity to exchange knowledge and experiences between people involved in this area, employed in various business and academic institutions. As a result, the domestic science conference on Computer Networks (CN) was initiated as an annual event dedicated to the academic and industry communities in Poland.

The conference was established at the Faculty of Automatic Control, Electronics and Computer Science at the Silesian University of Technology in Gliwice and the first five editions of the conference took place in the halls of this faculty. The cycle of these initial editions was approximately annual but not strictly constant. It depended on various conditions, e.g., the fact that the event was new and not widely known. From year to year, the conference grew in popularity and recognition and its range gradually evolved from local to nationwide. In 1999, the conference was organized as an external event for the first time. Each of the next editions was organized outside the university.

From the very beginning, for many years, Prof. Andrzej Grzywak together with Ms. Halina Węgrzyn worked actively on the conference organization, becoming the main stem of the Organizing Committee. In 2007, Prof. Grzywak retired and Dr. Piotr Gaj became the chair of the Organizing Committee while Prof. Andrzejń Kwiecie became the head of the Technical Program Committee. In 2008, the form of the conference was transformed from domestic into international with English as the official language. This gave guests from abroad an opportunity to participate in the CN conference.

In 2009, the conference proceedings were published in Springer's CCIS series for the first time, as volume 39. Since then, the conference proceedings have been published in this series each year.

For many years, among the co-organizers, technical co-sponsors, and partners has been the Section of Computer Networks and Distributed Systems belonging to the Committee on Informatics of the Polish Academy of Sciences (PAN), IEEE Poland Section, and the International Network for Engineering Education and Research (iNEER).

25th International Science Conference *Computer Networks*

This year, the 25th edition of the conference took place. Hence, the scientific conference on Computer Networks has been present on the conference market for a quarter of a century, giving attendees a chance to meet people, discuss valid and important issues, as well as disseminate research results in the proceedings published in a significant and recognized book series. Over the past 25 years of the conference's history, all important topics related to computer network development have been discussed at the conference and major breakthroughs in this area have been deliberated. Thus, we believe that the event has had a significant contribution to the global pool of achievements in this domain.

Computer networks are still the main means for transferring data among distributed nodes of all computer systems currently available. Thus, this area of research and applications is up to date and very important for current industrial and social activities, as well as for the needs of the future. Computer network-related techniques and technologies are not spectacular at their base but without them many spectacular services would be unavailable.

For this jubilee edition of the CN conference, almost 90 papers were submitted. To maintain the high quality of the conference proceedings, only 34 carefully selected papers have been published in this book. Each paper was reviewed by three independent reviewers in a double-blind process. Each of four chapters includes highly stimulating studies that may interest a wide readership.

- Computer Networks
  This chapter contains 12 papers. They refer to general networking issues, such as architecture design, analyzing, modeling, and programming computer networks, issues related to networked systems, their operations and management, Internet networks, cooperative and cognitive networks, issues related to quantum technology, as well as multimedia networks.
- Teleinformatics and Telecommunications
  This chapter contains five papers related to communication architecture, wireless systems, sensor and mobile networking, and isochronous communication.

– Queueing Theory
  This chapter contains eight papers. The papers refer to the theory of queues and queuing networks, as well as to network modelling and architecture design from the queueing theory point of view.
– Cybersecurity and Quality of Service
  This chapter contains nine papers that refer to cybersecurity issues in computer networks, as well as topics related to the quality of service domain and related innovative solutions.

On behalf of the Program and Organizing Committee of the CN Conference, we would like to express our gratitude to all authors for sharing their research results and for their assistance in producing this volume, which we believe is a reliable reference in the computer networks domain.

We also want to thank the members of the Technical Program Committee and all reviewers for their involvement and participation in the reviewing process.

If you would like to help us make the CN conference more attractive and interesting, please send us your opinions and propositions at cn@polsl.pl.

April 2018

Piotr Gaj
Grażyna Suchacka
Michał Sawicki
Andrzej Kwiecień

Organization

CN 2018 was organized by the Institute of Informatics of the Faculty of Automatic Control, Electronics and Computer Science, Silesian University of Technology (SUT) and supported by the Committee on Informatics of the Polish Academy of Sciences (PAN), the Section of Computer Networks and Distributed Systems in technical co-operation with the IEEE and consulting support of the iNEER organization.

His Magnificence Rector of Silesia University of Technology, the Mayor of Gliwice and Mayor of Katowice were the honorary patrons.

The official sponsor and partner of the CN 2018 conference was Wasko S.A. The companies 3Soft S.A. and Bombardier Inc. were conference partners.

Executive Committee

All members of the Executive Committee are from the Silesian University of Technology, Poland.

Honorary Member	Halina Węgrzyn
Organizing Chair	Piotr Gaj
Technical Volume Editor	Michał Sawicki
Technical Support	Aleksander Cisek
Technical Support	Jacek Stój
Office	Małgorzata Gładysz
Web Support	Piotr Kuźniacki

Co-ordinators

PAN Co-ordinator	Tadeusz Czachórski
IEEE PS Co-ordinator	Jacek Izydorczyk
iNEER Co-ordinator	Win Aung

Program Committee

Program Chair

Andrzej Kwiecień	Silesian University of Technology, Poland

Honorary Members

Win Aung	iNEER, USA
Adam Czornik	Silesian University of Technology, Poland
Bogdan M. Wilamowski	Auburn University, USA

Technical Program Committee

Davide Adami	University of Pisa, Italy
Wessam Ajib	University of Quebec at Montreal, Canada
Olumide Akinwande	Imperial College London, UK
Iosif Androulidakis	University of Ioannina, Greece
Tülin Atmaca	Institut National de Télécommunication, France
Rajiv Bagai	Wichita State University, USA
Jiří Balej	Mendel University in Brno, Czech Republic
Alexander Balinsky	Cardiff University, UK
Zbigniew Banaszak	Warsaw University of Technology, Poland
Thomas Bauschert	Chemnitz University of Technology, Germany
Robert Bestak	Czech Technical University in Prague, Czech Republic
Grzegorz Bocewicz	Koszalin University of Technology, Poland
Leoš Bohac	Czech Technical University in Prague, Czech Republic
Leszek Borzemski	Wrocław University of Technology, Poland
Juan Felipe Botero	Universidad de Antioquia, Colombia
Markus Bregulla	University of Applied Sciences Ingolstadt, Germany
Berk Canberk	Istanbul Technical University, Turkey
Amlan Chatterjee	California State University, USA
Ray-Guang Cheng	National University of Science and Technology, Taiwan
Erik Chromý	Slovak University of Technology, Slovakia
Yeon Ho Chung	Pukyong National University, South Korea
Andrzej Chydziński	Silesian University of Technology, Poland
Tadeusz Czachórski	Silesian University of Technology, Poland
Dariusz Czerwiński	Lublin University of Technology, Poland
Andrzej Duda	INP Grenoble, France
Alexander N. Dudin	Belarusian State University, Belarus
Peppino Fazio	University of Calabria, Italy
Max Felser	Bern University of Applied Sciences, Switzerland
Holger Flatt	Fraunhofer IOSB-INA, Germany
Jean-Michel Fourneau	Versailles University, France
Janusz Furtak	Military University of Technology, Poland
Rosario G. Garroppo	University of Pisa, Italy
Natalia Gaviria	Universidad de Antioquia, Colombia
Erol Gelenbe	Imperial College, UK
Roman Gielerak	University of Zielona Góra, Poland
Mariusz Głąbowski	Poznan University of Technology, Poland
Agustín J. González	Federico Santa María Technical University, Chile
Peter Gregorius	Beuth University of Applied Sciences, Germany
Sebastien Harispe	IMT Mines Alés, France
Faouzi Hidoussi	Corgascience Limited, Algeria
Edward Hrynkiewicz	Silesian University of Technology, Poland
Zbigniew Huzar	Wrocław University of Technology, Poland
Jacek Izydorczyk	Silesian University of Technology, Poland

Sergej Jakovlev	University of Klaipeda, Lithuania
Jürgen Jasperneite	Ostwestfalen-Lippe University of Applied Sciences, Germany
Krzysztof Juszczyszyn	Wrocław University of Science and Technology, Poland
Jerzy Klamka	IITiS Polish Academy of Sciences, Gliwice, Poland
Wojciech Kmiecik	Wrocław University of Science and Technology, Poland
Grzegorz Kołaczek	Wrocław University of Science and Technology, Poland
Ivan Kotuliak	Slovak University of Technology in Bratislava, Slovakia
Zbigniew Kotulski	Warsaw University of Technology, Poland
Demetres D. Kouvatsos	University of Bradford, UK
Stanisław Kozielski	Silesian University of Technology, Poland
Henryk Krawczyk	Gdańsk University of Technology, Poland
Udo R. Krieger	University of Bamberg, Germany
Mirosław Kurkowski	Cardinal Stefan Wyszynski University in Warsaw, Poland
Andrzej Kwiecień	Silesian University of Technology, Poland
Piotr Lech	West-Pomeranian University of Technology, Poland
Ricardo Lent	University of Houston, USA
Jerry Chun-Wei Lin	Harbin Institute of Technology, China
Wolfgang Mahnke	TE Industrial, Germany
Francesco Malandrino	Politecnico di Torino, Italy
Aleksander Malinowski	Bradley University, USA
Marcin Markowski	Wrocław University of Science and Technology, Poland
Przemysław Mazurek	West-Pomeranian University of Technology, Poland
Agathe Merceron	Beuth University of Applied Sciences, Germany
Jarosław Miszczak	IITiS Polish Academy of Sciences, Poland
Vladimir Mityushev	Pedagogical University of Cracow, Poland
Evsey Morozov	Petrozavodsk State University, Russia
Włodzimierz Mosorow	Lodz University of Technology, Poland
Sasa Mrdovic	University of Sarajevo, Bosnia and Herzegovina
Mateusz Muchacki	Pedagogical University of Cracow, Poland
Gianfranco Nencioni	University of Stavanger, Norway
Sema F. Oktug	Istanbul Technical University, Turkey
Michele Pagano	University of Pisa, Italy
Nihal Pekergin	University Paris-Est Creteil, France
Maciej Piechowiak	University of Kazimierz Wielki in Bydgoszcz, Poland
Piotr Pikiewicz	College of Business in Dabrowa Górnicza, Poland
Jacek Piskorowski	West Pomeranian University of Technology, Poland
Bolesław Pochopień	Silesian University of Technology, Poland
Oksana Pomorova	Khmelnitsky National University, Ukraine
Sławomir Przyłucki	Lublin University of Technology, Poland

Tomasz Rak	Rzeszow University of Technology, Poland
Stefan Rass	Alpen-Adria-Universität Klagenfurt, Austria
Przemysław Ryba	Wrocław University of Science and Technology, Poland
Vladimir Rykov	Russian State Oil and Gas University, Russia
Wojciech Rząsa	Rzeszow University of Technology, Poland
Dariusz Rzońca	Rzeszow University of Technology, Poland
Alexander Schill	Technische Universität Dresden, Germany
Artur Sierszeń	Lodz University of Technology, Poland
Mirosław Skrzewski	Silesian University of Technology, Poland
Adam Słowik	Koszalin University of Technology, Poland
Pavel Smolka	University of Ostrava, Poland
Tomas Sochor	University of Ostrava, Czech Republic
Maciej Stasiak	Poznań University of Technology, Poland
Janusz Stokłosa	Poznań University of Technology, Poland
Ioannis Stylios	University of the Aegean, Greece
Grażyna Suchacka	University of Opole, Poland
Wojciech Sułek	Silesian University of Technology, Poland
Zbigniew Suski	Military University of Technology, Poland
Bin Tang	California State University, USA
Kerry-Lynn Thomson	Nelson Mandela Metropolitan University, South Africa
Oleg Tikhonenko	Cardinal Stefan Wyszynski University, Poland
Ewaryst Tkacz	Silesian University of Technology, Poland
Homero Toral Cruz	University of Quintana Roo, Mexico
Mauro Tropea	University of Calabria, Italy
Leszek Trybus	Rzeszów University of Technology, Poland
Kurt Tutschku	Blekinge Institute of Technology, Sweden
Selda Uyanik	Istanbul Technical University, Turkey
Adriano Valenzano	National Research Council of Italy, Italy
Peter van de Ven	Eindhoven University of Technology, The Netherlands
Bane Vasic	University of Arizona, USA
Miroslaw Voznak	VSB-Technical University of Ostrava, Czech Republic
Krzysztof Walkowiak	Wrocław University of Technology, Poland
Sylwester Warecki	Intel, USA
Jan Werewka	AGH University of Science and Technology, Poland
Tadeusz Wieczorek	Silesian University of Technology, Poland
Lukasz Wisniewski	Hochschule Ostwestfalen-Lippe, Germany
Przemysław Włodarski	West Pomeranian University of Technology, Poland
Józef Woźniak	Gdańsk University of Technology, Poland
Hao Yu	Auburn University, USA
Grzegorz Zaręba	University of Arizona, USA
Piotr Zawadzki	Silesian University of Technology, Poland
Zbigniew Zieliński	Military University of Technology, Poland
Liudong Zuo	California State University, USA
Piotr Zwierzykowski	Poznań University of Technology, Poland

Referees

Davide Adami
Rajiv Bagai
Sebastian Bala
Jiří Balej
Zbigniew Banaszak
Thomas Bauschert
Robert Bestak
Grzegorz Bocewicz
Leoš Bohac
Juan Felipe Botero
Amlan Chatterjee
Ray-Guang Cheng
Erik Chromý
Yeon Ho Chung
Andrzej Chydziński
Dariusz Czerwiński
Adam Czubak
Peppino Fazio
Max Felser
Holger Flatt
Jean-Michel Fourneau
Rosario G. Garroppo
Natalia Gaviria
Erol Gelenbe
Roman Gielerak
Mariusz Głąbowski
Daniel Grzonka
Sebastien Harispe
Artur Hłobaż
Zbigniew Huzar
Jacek Izydorczyk
Sergej Jakovlev
Agnieszka Jakóbik
Jürgen Jasperneite
Anna Kamińska-Chuchmaa
Jerzy Klamka
Grzegorz Kołaczek
Ivan Kotuliak
Zbigniew Kotulski
Henryk Krawczyk
Mirosław Kurkowski
Andrzej Kwiecień
Piotr Lech
Zbigniew Lipiński
Marcin Markowski
Francesco Masulli
Przemysław Mazurek
Jarosław Miszczak
Vladimir Mityushev
Alexander Moiseev
Evsey Morozov
Włodzimierz Mosorow
Sasa Mrdovic
Mateusz Muchacki
Gianfranco Nencioni
Sema F. Oktug
Remigiusz Olejnik
Michele Pagano
Nihal Pekergin
Maciej Piechowiak
Piotr Pikiewicz
Jacek Piskorowski
Oksana Pomorova
Sławomir Przyłucki
Tomasz Rak
Stefan Rass
Stefano Rovetta
Przemysław Ryba
Wojciech Rząsa
Dariusz Rzońca
Alexander Schill
Olga Siedlecka-Lamch
Artur Sierszeń
Tomas Sochor
Janusz Stokłosa
Ioannis Stylios
Grażyna Suchacka
Wojciech Sułek
Zbigniew Suski
Bin Tang
Oleg Tikhonenko
Ewaryst Tkacz
Homero Toral Cruz
Mauro Tropea
Leszek Trybus
Kurt Tutschku
Peter van de Ven
Miroslaw Voznak
Sylwester Warecki
Jan Werewka
Tadeusz Wieczorek
Lukasz Wisniewski
Przemysław Włodarski
Józef Woźniak
Hao Yu
Krzysztof Zatwarnicki
Zbigniew Zieliński
Liudong Zuo
Piotr Zwierzykowski

Sponsoring Institutions

Organizer: Institute of Informatics, Faculty of Automatic Control, Electronics and Computer Science, Silesian University of Technology
Co-organizer: Committee on Informatics of the Polish Academy of Sciences, Section of Computer Networks and Distributed Systems

Honorary Patrons: His Magnificence Rector of Silesia University of Technology, Mayor of Gliwice, and Mayor of Katowice
Official Sponsor and Partner: Wasko S.A.
Official Partners: 3Soft S.A. and Bombardier Inc.

Technical Partner

Technical Co-sponsor: IEEE Poland Section
Conference Partner: iNEER

Contents

Computer Networks

Teleinformatics and Telecommunications

Queueing Theory

Computer Networks

The Method of Determining the Optimal Communication Structure

Jan Chudzikiewicz(✉) and Tomasz Malinowski(✉)

Faculty of Cybernetics, Military University of Technology,
ul. Gen. Witolda Urbanowicza 2, 00-908 Warszawa, Poland
{jan.chudzikiewicz,tomasz.malinowski}@wat.edu.pl
http://www.wat.edu.pl/

Abstract. In the paper, the problem of determining the optimal communication structure (communication routes) for wireless or wired data communication networks with predefined structure of usable communication links was considered. The procedure of determining such structure, which is based on identifying and comparing of selected characteristics of the dendrites describing the network nodes and communication links, was developed. The correctness of the method has been confirmed by results obtained by comparative simulation studies of different communication substructures (dendrites). Simulation tests were prepared and implemented in Riverbed Modeler environment.

Keywords: Data communication networks
Optimal communication structure · Internet of Things

1 Introduction

Determining the optimal communication structures is the issue raised in many of research works focused on improving the reliability, performance, and usability of transmission systems (multiprocessor systems, military computer networks - wired and wireless networks of stationary or mobile nodes). A lot of research focuses on the problems with networks of the hypercube structure and on the problems with location and relocation of network resources ([1–7]). In the era of IoT (Internet of Things), results of research on "energy-efficient" communication structures, prolong the life time of the network with battery-powered nodes, are particularly important ([8–10]). The work is focused on developing new routing protocols, effective medium access protocols, selecting of nodes collecting and processing information from other nodes, constituting the catchment information nodes (sink nodes), meeting points nodes (rendezvous points) or acting as coordinator nodes ([10–14]).

In this paper we introduce a centralized procedure for determining the optimal communication structure. The result of the procedure is the shortest path tree with the root in the specific node. We abstract from considering the role this node can play (server, coordinator, collector of information from other nodes). We assume

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 3–12, 2018.
https://doi.org/10.1007/978-3-319-92459-5_1

for example, as many articles on the same topic, that indication and use "our" optimal communication structure in a network of wireless nodes, will result in reducing the number of packets generated by the nodes and exchanged between nodes, thereby limiting power consumption and increase the life of network.

Our procedure for determining the optimal communication structure is centralized and we hope that it is universal, on the assumption that we will be able to assess, based on various parameters (bandwidth of transmission channels, delays, packets lost, etc.), the functioning of the network (wireless or wired) at a given moment. We have assumed that we will get information about routing and quality of communication connections (from sensors testing communication channels). On this basis, we will be able to give an optimal (probably suboptimal) communication structure resulting from the use of the best communication routes (dendrite of the best routes). These routes can then be entered (distributed) into the configuration files of network devices. In our opinion, a centrally determined suboptimal communication structure, considering not only the number of hops from the source to the destination (in our procedure only this characteristic has been taken into account), but also other important parameters and regardless of the routing protocol used (whether for wired or wireless networks) will have a positive effect on the functioning of the network (minimizing delays, energy consumption by communication nodes, etc.). We also assumed that the analyzed network is rather small (cluster of whole network) and in our assumptions it's size is limited to ten nodes.

The paper is organized as follows. In Sect. 2, basic definitions were introduced. The calculation of attainability for exemplary structures were presented. In Sect. 3, the proposal of the method and the algorithm of determining optimal communication structure was presented. In Sect. 4, the results of simulation tests for proposed method verification were described. In Sect. 5, some concluding remarks were presented.

2 The Basic Definitions

Definition 1. *The structure of a network is described by coherent ordinary graph* $G = \langle E, U \rangle$ *(*E *– set of network nodes,* U *– set of bidirectional communication link).*

Let $d\,(e, e'\,|G\,)$ be the distance (the number of hops) between nodes e and e' in a coherent graph G, that is the length of the shortest chain (in the graph G) connecting node e with the node e'.

Let $r\,(e|G) = \underset{e' \in E(G)}{max}\, d\,((e,\, e')\,|G\,)$ be the radius of a node, and $D\,(G) = max\,\{d\,(e', e''\,|G\,) : \{e', e''\} \subset E\,(G)\}$ be the diameter of a graph G. The radius of node e is the largest number of hops from node e to any other node of G. The diameter of G is the largest radius in G.

Denote by $E^{(d)}\,(e|G) = \{e' \in E\,(G) : d\,(e, e'\,|G\,) = d\}$ for $d \in \{1, \ldots, D\,(G)\}$, and by

$$\varsigma\,(e\,|G) = (\varsigma_1\,(e\,|G\,), \ldots, \varsigma_r\,(e\,|G\,)) \tag{1}$$

for $\varsigma_d\,(e\,|G\,) = \left|E^{(d)}\,(e\,|G\,)\right|$.

Definition 2 [6]. *Let* $\varphi(e|G) = \sum_{e' \in E(G)} d(e, e'|G)$ *for* $e \in E(G)$ *be attainability of the node* e *in the network* G *and* $\Phi(G) = \sum_{e \in E(G)} \varphi(e|G)$ *be attainability of the network* G.

Using (1) we have

$$\varphi(e|G) = \sum_{d=1}^{r(e|G)} d_{Sd}(e|G).$$

Definition 3 [7]. *Let* $T = \langle E, U^* \rangle$ *be the dendrite i.e. such coherent acyclic partial graph of* G *that:*

$$\exists \langle e', e'' \rangle \in U \Longrightarrow \langle e', e'' \rangle \in U^* \Longleftrightarrow$$

$$\Longleftrightarrow [(d(e_i, e') \neq d(e_i, e'')) \wedge d(e', e'') = 1]$$

for $r(e_i) =_{e \in E(G)}^{min} r(e)$.

Denote by $T(e)$ for $e \in E(G)$ the set of all possible dendrites determined for node e.

Let $\Phi_{\min}(T(e)) =_{t \in T(e)}^{min} \sum_{e \in E(t)} \varphi(e|t)$ be the value of minimal attainability for $T(e)$.

The dendrite T is a base to determine the communication structure of G. The algorithm for determined dendrite T is presented in [7]. The method and the algorithm for determined the optimal communication structure based on the dendrites determination is presented in Sect. 3.

3 The Method and the Algorithm for Determining the Optimal Communication Structure

The method consists of fourth stages. In the first stage for G, as the first node we choose a node that $r(e|G) =_{e' \in E(G)}^{min} r(e')$ or $\varphi(e|G) =_{e' \in E(G)}^{min} \varphi(e')$. In the second stage for the chosen node we determine all possible dendrites $T(e)$ for $e \in E(G)$. In the third stage, for each dendrite determined in the second stage, the value of attainability $\Phi(T(e))$ is calculated. In the fourth stage, based on the attainability calculated in the third stage, the dendrite T, which is an optimal communication structure satisfying the condition $\Phi_{\min}(T(e))$ is determined. Based on the presented method the algorithm for determining the optimal communication structure was developed.

The algorithm for determining the optimal communication structure.

Step 1. *(first stage)*

Calculate $r(e|G)$ and $\varphi(e|G)$ for $e \in E(G)$.

Choose a node $e^* \in E(G)$ such that $r(e^*|G) =_{e' \in E(G)}^{min} r(e')$ or

$\varphi(e^*|G) =_{e' \in E(G)}^{min} \varphi(e')$.

Selected node e_i will be a central node of G.

Step 2. *(second stage)*

For the chosen node e^* determine set $T(e^*)$, wherein $r(e^*|T) = r(e^*|G)$.

Step 3. *(third stage)*

Calculate $\Phi(t(e^*))$ for $t \in T(e^*)$.

Step 4. *(fourth stage)*

Choose a dendrite t such that $\Phi(t) =_{t' \in T(e^*)}^{min} \Phi(t')$.

Step 5.

The end of the algorithm.

For illustrating the algorithm's operation, consider the following example.

Example 1. Let the structure G_1 (presented in Fig. 1) be the basis for determining the optimal communication structure.

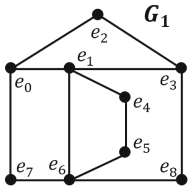


Fig. 1. An exemplary communication structure G_1

In the first step of the presented algorithm, the radius and the attainability for each node of G_1 must be calculated. In the Table 1 the results of calculating radius (A) and attainability (B) of G_1's nodes are presented.

Table 1. The results of determining the radius (A) and the attainability (B) of G_1

A		B					
$e \in E(G_1)$	$r(e\|G_1)$	$e \in E(G_1)/d(e,e'\|G_1)$	**1**	**2**	**3**	**4**	$\varphi(e,G_1)$
e_0	3	e_0	3	3	2	0	15
e_1	2	e_1	4	4	0	0	12
e_2	4	e_2	2	3	2	1	18
e_3	3	e_3	3	3	2	0	15
e_4	3	e_4	2	3	3	0	14
e_5	4	e_5	2	3	2	1	18
e_6	3	e_6	4	3	1	0	13
e_7	4	e_7	2	4	2	0	16
e_8	4	e_8	2	4	2	0	16

Next, the node with a minimal radius or (if nodes with a minimum radius value is more) the node with a minimal attainability is chosen. In the case of G_1, the node with the minimal radius, and minimal attainability is node e_1. In the next step, for chosen node e_1, the algorithm determines the set $T(e_1)$ (presented in Fig. 2) of possible dendrites and calculates the attainability (presented in Table 2) for all of them.

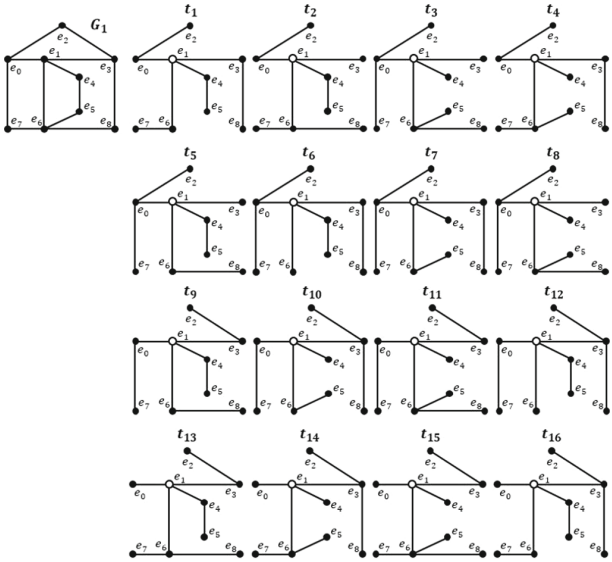


Fig. 2. The set $T(e_1)$ of possible dendrites of the e_1 node

The algorithm, as the optimal communication structure T_{OPT} for G_1, choses the dendrite t_{15} shown in Fig. 3.

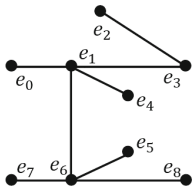


Fig. 3. The optimal communication structure T_{OPT} for G_1

In Table 3 the results of calculating radius (A) and attainability (B) of T_{OPT} are presented.

Table 2. The attainability for dendrites of set $T(e_1)$

$\varphi(e)/T(e_1)$	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}	t_{12}	t_{13}	t_{14}	t_{15}	t_{16}
$\varphi(e_0)$	17	17	15	17	15	15	15	15	17	17	17	17	19	19	19	19
$\varphi(e_1)$	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12
$\varphi(e_2)$	24	24	22	24	22	22	22	22	24	22	20	22	24	22	24	22
$\varphi(e_3)$	17	19	19	17	20	17	17	19	17	15	17	15	17	15	17	15
$\varphi(e_4)$	17	18	19	19	17	17	19	19	17	19	19	17	17	19	19	17
$\varphi(e_5)$	24	24	22	22	24	24	24	22	24	24	22	24	24	22	20	24
$\varphi(e_6)$	17	15	15	15	17	19	17	15	17	17	15	19	15	15	13	17
$\varphi(e_7)$	24	22	22	22	22	22	22	22	24	24	24	20	22	22	20	24
$\varphi(e_8)$	24	22	22	24	24	24	24	22	24	22	22	22	22	22	20	22
$\Phi(T(e_1))$	176	173	168	172	173	172	172	168	176	172	168	168	172	168	164	172

Table 3. The radius (A), and the attainability (B) of T_{OPT}

A		B					
$e \in E(T_{OPT})$	$r(e\,\vert T_{OPT})$	$e \in E(T_{OPT})$ / $d(e,e'\,\vert T_{OPT})$	**1**	**2**	**3**	**4**	$\varphi(e,T_{OPT})$
e_0	3	e_0	1	3	4	0	19
e_1	2	e_1	4	4	0	0	12
e_2	4	e_2	1	1	3	3	24
e_3	3	e_3	2	3	3	0	17
e_4	3	e_4	1	3	4	0	19
e_5	4	e_5	1	3	3	1	20
e_6	3	e_6	4	3	1	0	13
e_7	4	e_7	1	3	3	1	20
e_8	4	e_8	1	3	3	1	20
			$\Phi(T_{OPT})$				164

4 The Results of Simulation Studies

Simulation studies have been realized in Riverbed Modeler environment. The verification of method proposed and described in Sect. 3 was carried out in a wired network environment (due to the rapid and easy modeling of many network structures).

In this subsection, we present results for the procedure verification for $T(e_1)$ set of dendrites. From section Sect. 3 we know that the best communication structure for these set is t_{15} (T_{OPT} for G_1).

The conducted simulation tests certainly do not serve to verify the design of any network. The tests were to authenticate the procedure for determining the optimal communication structure. We decided that the simulation model does not have to be or even should not be a model fully adequate to the real network. We assumed that it would be acceptable to design any communication

system with the minimal number of network services. We set the LAN model with an unspecified shape of network traffic and the selected http service. In our case, when assessing the impact of the routing method (corresponding to the chosen subgraph) on how the chosen service works, we decided that it would be the best solution, with transparent results. Therefore, we only took care of the homogeneity of links and network devices. What was important for us is that in the randomness of generating network streams and the randomness of client-server connections, the simulation results confirm the correctness of the theoretical arguments.

All dendrites were modeled as computer networks with routers with attached LAN segment and servers. Routers play a role of a dendrite's node. For example, the network topology of the single scenario (single dendrite) was shown in the Fig. 4.

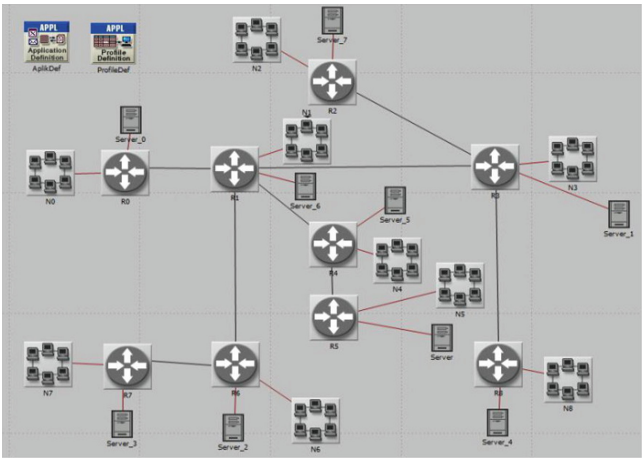


Fig. 4. Network topology for single simulation scenario

Workstations within LAN segments (ten workstations in each segment) were functioning as the server's clients. The basic server's service was http server (quite enough to carry out research), with standard application [15]. Communication links bandwidth was set to 1,5 Mb/s.

We performed some tests with other settings of network traffic generators and the number of tcp clients in each network segments (to saturate network links), but the obtained results for the characteristics chosen by us (number of hops and delay for tcp transmission) were comparable, i.e. have indicated the communication structure chosen by the procedure, so we gave up the detailed description of each experiments by selecting a representative one.

Selected results (for set $T(e_1)$ of dendrites), confirming the correctness of the procedure for determining the optimal communication structure are shown in the figures below.

We used two important characteristics (in our opinion sufficient at this stage of research). The first of them was global average Number of Hops (representing an average number of IP hops taken by data packets reaching at a destination node) and the second, average TCP Delay (representing delay of TCP packets. This value is measured from the time an application data packet is sent from the source TCP layer to the time it is received by the TCP layer in the destination node for all connections).

We knew which communication structure for each set of dendrites is optimal and we were looking for confirmation of the correctness of calculations performed earlier. The collected results looked like this from Fig. 5. Figure illustrates average Number of Hops for the best dendrite t_{15} (the lowest drawn line) against results for selected three other dendrites.

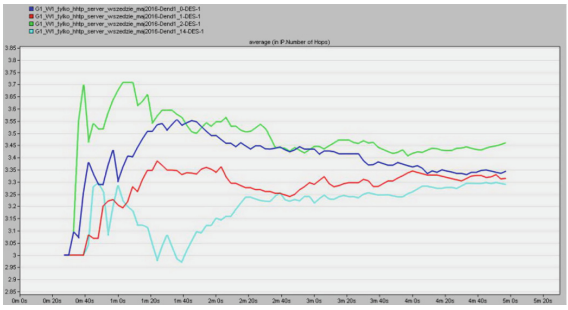


Fig. 5. Average Number of Hops for selected dendrites

Complete results, in the form of a bar chart, for all dendrites from set $T(e_1)$ are presented in Fig. 6. The highlighted bar (No 15) refers to the best structure t_{15}.

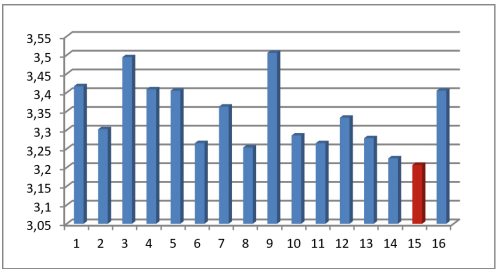


Fig. 6. Average Number of Hops for all dendrites of $T(e_1)$

Similarly, for all dendrites from the selected set, average *TCP Delay* was measured. The results are presented in Fig. 7.

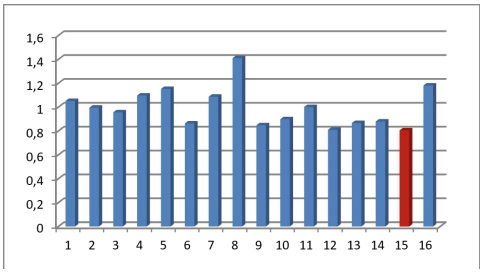


Fig. 7. Average TCP delay (in seconds) for TCP-based services

Results of simulation studies (we expected the lowest values of Average Number of Hops and Average TCP delay for TCP-based services) confirmed the correctness of the developed method of determining the optimal communication structure, determined analytically in section Sect. 3.

5 Summary

Correctness of developed method and its usefulness for determining optimal communication structure was confirmed by simulation tests. In analytical calculations two essential characteristics, number of hops and attainability, were used. According to our expectations, in simulation tests you could also use other characteristics, reflecting real condition of the network (in our study we used only TCP delay, but it was quite enough to confirm the correctness of analytical argumentations). The authors believe that the developed method can be used to determine the optimal structure also in wireless networks (for example to extend its life or for the efficient collection of information from wireless nodes). In the nearest future we plan to investigate the impact of such a network monitoring procedure with a periodically scheduled routing plan on how a specific network of wireless nodes works.

At this stage, for a small communication structure, complexity of the developed algorithm was not calculated. Our observations show that the efficiency of the algorithm depends on the number of nodes, the structure diameter and the radius of the node for which dendrites were determined. Currently, calculations are performed on a PC and it does not matter the memory usage and CPU load. We notice the need to modify the procedure (to be fast and not burdening the system) if it is going to be implemented by a wireless network node.

References

1. AlBdaiwia, B.F., Bose, B.: On resource placements in 3D tori. J. Parallel Distrib. Comput. **63**, 838–845 (2003)
2. AlBdaiwia, B.F., Bose, B.: Quasi-perfect resource placements for two-dimensional toroidal networks. J. Parallel Distrib. Comput. **65**, 815–831 (2005)

3. Bae, M.M., Bose, B.: Resource placement in torus-based networks. IEEE Trans. Comput. **46**(10), 1083–1092 (1997)
4. Imani, N., Sarbazi-Azad, H., Zomaya, A.Y.: Resource placement in Cartesian product of networks. J. Parallel Distrib. Comput. **70**, 481–495 (2010)
5. Moinzadeh, P., Sarbazi-Azad, H., Yazdani, N.: Resource placement in cube-connected cycles. In: The International Symposium on Parallel Architectures, Algorithms, and Networks, pp. 83–89. IEEE Computer Society (2008)
6. Chudzikiewicz, J., Zieliński, Z.: On some resources placement schemes in the 4-dimensional soft degradable hypercube processors network. In: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds.) Proceedings of the Ninth International Conference on Dependability and Complex Systems DepCoS-RELCOMEX. June 30 – July 4, 2014, Brunów, Poland. AISC, vol. 286, pp. 133–143. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07013-1_13
7. Chudzikiewicz, J., Malinowski, T., Zieliński, Z.: The method for optimal server placement in the hypercube networks. In: Proceedings of the 2015 Federated Conference on Computer Science and Information Systems, ACSIS, Vol. 2, pp. 947–954 (2015). https://doi.org/10.15439/2014F159
8. Niewiadomska-Szynkiewicz, E., Kwaśniewski, P., Windyga, I.: Comparative study of wireless sensor networks energy-efficient. J. Telecommun. Inf. Technol. (2009)
9. Jindal, A., Liu, M.: Networked computing in wireless sensor networks for structural health monitoring. In: IEEE/ACM Transactions on Networking (TON) (2012)
10. Brindha, L., Muthaiah, U.: Energy efficient mobile sink path selection using a cluster based approach in WSNs. Int. J. Innov. Res. Comput. Commun. Eng. **3**(3) (2015)
11. Erzin, A.I., Plotnikov, R.V.: Using VNS for the Optimal synthesis of the communication tree in wireless sensor networks. Electron. Notes Discrete Math. **47**, 21–28 (2015)
12. Rekha, G., AjeethaKumari, A.S.: High data aggregation in wireless sensor networks using Rendezvous-Drina. Int. J. Emerg. Technol. Adv. Eng. **4**(6), 139–144 (2014)
13. Ghotra, A., Soni, N.: Performance evaluation of ant colony optimization based rendezvous leach using for mobile sink based WSNs. Int. J. Eng. Res. Dev. **11**(07), 43–49 (2015)
14. Baby, S., Soman, M.: Rendezvous based techniques for energy conservation in wireless sensor networks - a survey. J. Netw. Commun. Emerg. Technol. (JNCET) **3**(3) (2015)
15. Sethi, A.S., Hnatyshin, V.Y.: The Practical OPNET User Guide for Computer Network Simulation. Chapman and Hall/CRC, Boca Raton (2012)

Researching Measured and Modeled Traffic with Self-similar Properties for Ateb-Modeling Method Improvement

Olga Fedevych[1(✉)], Ivanna Dronyuk[1], and Danylo Lizanets[2]

[1] ACS Department, Lviv Polytechnic National University,
12 Bandera Street, Lviv, Ukraine
olha.fedevych@gmail.com, ivanna.droniuk@gmail.com
[2] Division of Microengineering and Photovoltaics,
Wrocław University of Science and Technology,
27 Wybrzeze Wyspiańskiego, Wrocław, Poland
danylo.lizanets@pwr.edu.pl
http://www.lp.edu.ua/asu
http://www-old.wemif.pwr.wroc.pl/zmf/

Abstract. In this paper, improvements in the traffic behavior modeling method based on the consideration of traffic self-similarity properties was proposed. Two methods for calculating the Hurst parameter were used: R/S plot method and V/S analysis method. The research was carried out on real traffic network data, collected from LPNU ACS Department. The selected methods were implemented in the corresponding software. The work of the methods was illustrated in experimental calculations. Also, the results was shown in the form of both tables and plots.

Keywords: Computer network traffic · Ateb-function
Traffic modeling · Self-similar properties · Hurst parameter

1 Introduction

One of the biggest challenges nowadays is the gradual shift from the Internet of Devices concept to the new Internet of Things concept, given the aspirations of those who developed these concepts to boost their user-friendliness and the growing demand for dynamic scalability of networks that differ by scale, type or functionality. Also, the revolutionary influence on traffic parameters changing has an alteration of its internal structure. Based on today's Internet research, provided by Cisco, mobile traffic on 75% consists of the video traffic [1].

According to the forecasts of the same company, the share of video content in the future will only be increasing. Another factor that significantly influences the structure of traffic is the widespread introduction of cloud technologies. The cloud computing paradigm has become the foundation of the modern economy by offering subscription-based services anytime, anywhere with services paid by

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 13–25, 2018.
https://doi.org/10.1007/978-3-319-92459-5_2

users. The transition of many technologies to distributed computing, software-managed networks and data processing at the network level in nodes of network equipment are also among the modern trends that open up new opportunities for traffic management [2,3].

At the same time, computer networks have certain disadvantages, such as: the complexity of network management, the high cost of network equipment, the lack of efficient use of the channel through the transmission of a large amount of information to manage the network instead of useful traffic. Multimedia traffic samples were selected for the research.

The purpose of this development is to expand the functionality of the previously [4] proposed method of modeling the behavior of traffic based on the consideration of self-similar properties of traffic. In this article, under the term of traffic modeling (forecasting), the conception of the process of collecting data for real traffic and based on these data and the corresponding formulas of the mathematical model, the calculation of the following traffic values was considered.

2 Simulation of Traffic Behavior for Model Construction

One of the most pressing problems of traffic research is adequate consideration of its features. Studies show that the traffic of modern telecommunication networks has a special structure that does not allow to use classical methods to design network equipment based on Markov models and Erlang's formulas [4]. This is the effect of self-similarity of traffic, which leads to the ripples of its receipt. This phenomenon significantly degrades the characteristics (increases losses, delays, jitter) while passing self-similar traffic through the network equipment, so the study of this indicator will predict and reduce the impact of such undesirable factors.

Research of various types of network traffic over the past fifteen years has shown that network traffic is self-similar or fractal in its nature. It follows from this that the classical methods that are used for modeling of network systems based on the use of Poisson flows [5], do not give a complete and accurate picture of what is happening online. In [4] a mathematical model for modeling traffic behaviour in computer networks based on differential equations describing the oscillatory motion was proposed.

In addition, self-similar traffic has a special structure, stored at multiple scaling. When passing traffic on the network, as a rule, there is a certain number of pulsations with a relatively small average traffic level. In practice, this is implemented in such a way that packets, at high speed of their network movement, arrive at the site not separately, but as a whole package, which can lead to their losses due to limited buffer, calculated according to classical techniques.

Many contemporary researchers note [6] that combining of traffic from multiple variable ON/OFF sources increases self-similar properties of traffic. Traffic becomes highly autocorrelated with long-term dependence. The unification of a large number of data sources is characterized by a syndrome of infinite dispersion and, as a result, gives a self-similar unified network traffic that seeks for a

fractal Brownian motion. In addition, the research of various sources of traffic indicates that the very variable behavior of traffic is a function inherent in the client/server architecture [7].

There is no single causative factor that causes self-similarity. Different correlations that exist in self-similar network traffic and that take effect on different time scales may occur for various reasons, changing the characteristics at specific time scales.

3 Methods of Traffic Self-similar Properties Determination

To analyze the self-similar properties of traffic, the Hurst coefficient was selected. At the same time, the Hurst method, being robust, can reveal such properties as clustering, the tendency to be in the direction of the trend, strong aftereffect, separate memory, rapid change of sequential values, fractality, the presence of periodic and nonperiodic cycles, the ability to distinguish "stochastic" and "chaotic" nature of noise in the statistical data.

The value of the Hurst coefficient can be found in several methods [8–10]:

1. Variance method. Logarithmic selective dispersion in comparison with the level of aggregation should be a straight line with a slope of more than -1.
2. R/S plot method. The logarithmic sample of R/S statistics compared with the number of points in the aggregated series is a straight line with a slope.
3. Absolute moments method. The aggregated series $X(m)$ determines the use of different sizes of blocks m.
4. Variance of residuals method. Logarithmic aggregation in comparison with the average dispersion level of the residues should be a straight line with a slope of $H/2$.
5. Abry-Veitch estimation method. To estimate the H parameter, the energy of the series on different scales is used.
6. Whittle estimation method is based on the minimization of the likelihood function, which applies to the time series period, gives the assessment of H and dependence on the confidence interval.
7. V/S analysis, described in detail in [11].

All seven mentioned methods were tested for realization of scientific researches. During the computational experiments, R/S plot method and V/S analysis were selected as the best methods for determining the proximity of the Hurst coefficient for modeled and real traffic. These methods were chosen among others due to the ease of their program implementation.

In addition to the key contribution of Hurst to the development of the theory of R/S-method and its application in practice, a significant role was played by Mandelbrot [12]. According to this method, not the data that compose a dynamic time series are analyzed, but the scale of the amount of deviations of these data from the mean arithmetic, normalized by dividing on standard deviation. The sum of these deviations is calculated for different periods of time

(or for different number of successive observation points), which act as a scale of measurement.

The main difference between R/S estimation method and other existing statistical methods for analyzing of time series is that this method includes the time direction in its analysis, while other known methods for this time are invariant.

The Hurst H coefficient is described by the empirical relation

$$\frac{R}{S} = N^H \tag{1}$$

where R – the range of deviations values of the series x, S – standard deviation x, N – number of observations.

Express the Hurst H factor as:

$$H = \frac{\log \frac{R}{S}}{\log N} \tag{2}$$

Let $\bar{x}(N)$ – average value of random variable

$$\bar{x}(N) = \frac{1}{N} \sum_{i=1}^{N} x(t_i) \tag{3}$$

The standard deviation is determined from the formula

$$S(N) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} [x(t_i) - \bar{x}(N)]^2} \tag{4}$$

Denote by the following formula the accumulated deviation of the values of the random variable $x(t)$ from its mean value $\bar{x}(N)$ for time t:

$$X(t, N) = \sum_{u=1}^{t} [x(u) - \bar{x}(N)] \tag{5}$$

Difference between maximum and minimum values $X(t, N)$ is called a scope

$$R(N) = \max X(t, N) - \min X(t, N) \tag{6}$$

where $1 \leq t \leq N$.

Calculated deviations (4), (5) are substituted in the formula (1) and there is a Hurst coefficient.

Two methods for calculating the Hurst parameter are implemented in the developed software module. Conducted calculations of the Hurst parameter are implemented for the real traffic values and the modeled traffic values based on the R/S method and the V/S method described in detail in [11,12].

4 Development of Software Solution

This work envisaged the creation of two separate software solutions that collectively carried out modeling and traffic analysis in order to model its behavior. First, a solution for analysis of traffic samples and the implementation of the short-term behaviour modeling method was developed, followed by a separate module for analyzing and calculating the Hurst parameter in the collected traffic data.

To carry out these studies, an additional software module has been implemented to calculate the Hurst parameter for different time intervals, samples of traffic, as well as to calculate the Hurst parameter for pre-recorded modeled values of the short-term behaviour modeling method of traffic values.

Figure 1 shows the main window of the software module for calculating the Hurst parameter.

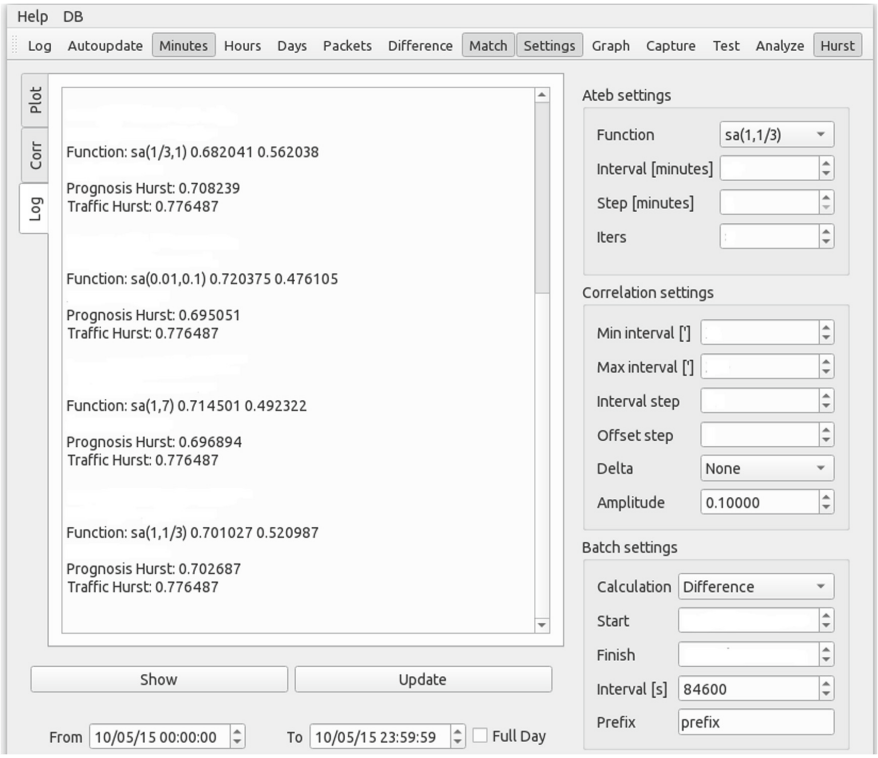


Fig. 1. Program module interface for Hurst parameter calculation

5 Analysis of Experimental Calculations

There is an ability to open captured traffic in the analyzer program, analyze and calculate the properties of the traffic. Opening of pcap files does not require much resources since the reading of the file occurs in the thread, though opening of the same file in other programs may require a lot of resources.

To test the traffic analysis software, data was captured in two different programs - Wireshark and CommView. Data capture was carried out in different places and different devices: a wireless adapter, a network card, a virtual network adapter.

The results were analyzed by a software solution for traffic analysis. It was established that since the captured data is exported to the standardized format of the pcap file, there were no errors in the reading. Traffic properties were determined correctly.

Developed software was also tested on different machines with different operating systems. After the testing, the software continued to work in as stable and predictable manner as before.

5.1 Description of Conducted Experiments

Traffic was captured at the point where it had already been merged and sent (received) from the Internet. The network of the ACS department contains about 20 working computers, loaded on average from 8:30 to 17:30; about 4 computers are loaded until 21:00, and 3 computer classes with 32 workstations are loaded on average from 8:30 to 16:00. The measurements were carried out during the month on each working day from 9.00 to 13.00 o'clock, at an average data transfer rate of 150 Mbit/sec. However, for visualization of the conducted research, three days in a row were randomly chosen.

The collected data was analyzed by the developed software solution in terms of traffic; the results of the analysis are shown in Table 1.

Duration of data collection differs in samples, which may slightly affect the Hurst coefficient. The simulation time is chosen identically to the time of data capture on a network.

In the previously proposed method of modeling, the property of the alternating period of the Ateb-functions was used to select modeling parameters.

In [13] is proved that the Ateb-cosine and the Ateb-sine are periodic with period $2\Pi(m,n)$, where $\Pi(m,n)$ is shown by the formula

$$\Pi(m,n) = \frac{\Gamma\left(\frac{1}{n+1}\right)\Gamma\left(\frac{1}{m+1}\right)}{\Gamma\left(\frac{1}{n+1}+\frac{1}{m+1}\right)} \tag{7}$$

where $\Gamma(\frac{1}{n+1}), \Gamma(\frac{1}{m+1})$ – gamma function.

For the modeling, the parameters n, m were selected in such a way that the modeling period corresponded to the period of real traffic. In order to apply the self-similarity property to improve the traffic behavior modeling, the Hurst parameter for Ateb-functions in the half-period was calculated by the formula (7). These calculations are presented in Table 1. Parameter t - modeling time, s.

Table 1. Dependency of Hurst parameter from Ateb-sine parameter

Function type	Hurst parameter value $H(m, n)$
sa(1, 1, t)	0.861452
sa(1/7, 3, t)	0.884723
sa(1/5, 1, t)	0.873587
sa(1/3, 1, t)	0.875528
sa(0.01, 0.1, t)	0.823017
sa(1, 7, t)	0.879125
sa(1, 1/3, t)	0.855861

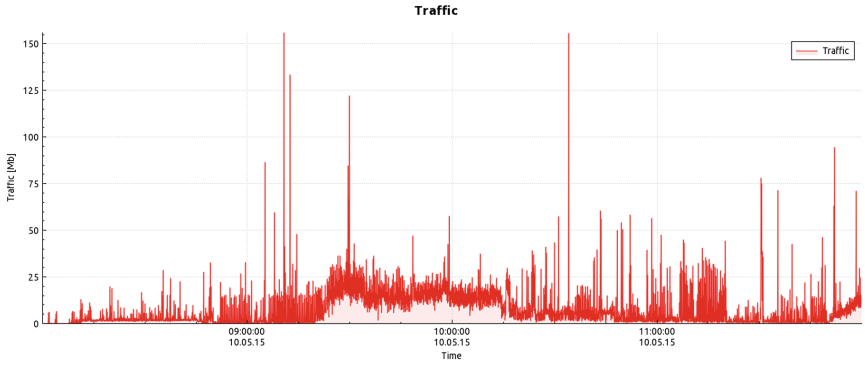


Fig. 2. The example of investigated traffic sample

Not only the proximity of the period of real and modeled traffic, but also the proximity of the Hurst parameter as the second factor are taken into account when choosing the values of the parameters n, m, when implementing modeling in the improved method (Fig. 2).

$$\min_{n,m} |H(m, n) - H_{tr}| \to 0 \tag{8}$$

where $H(m, n)$ – Hurst parameter for Ateb-function, H_{tr} – Hurst parameter for real traffic.

The traffic modeling is implemented using the method with the addition of the delta function after the parameters n, m are chosen with the consideration of the period and self-similarity using the formula (8).

5.2 Evaluation and Analysis of Computational Results

The obtained data was analyzed and the Hurst coefficient was obtained. The Hurst coefficient is consistently greater than 0.5 for all collected and simulated traffic data, which indicates a self-similarity effect appearance.

The Hurst coefficient (H) is a measure of the stability of a statistical phenomenon or measure of the duration of the long-term dependence of the process. The closer the value of H is to 1, the higher the degree of stability of long-term dependence and self-similarity is.

Thus, in the follow-up of the long-term observations, there is the possibility to learn the tendency of traffic and to be able to model network traffic load. If $0 \leq H \leq 0.5$, then the collected data is trend-resistant, if $0.5 \leq H \leq 1$, the series is the self-similar and trend-stable and thus the tendency of its changes can be predicted.

The calculation were carried out for the R/S method (Tables 1, 2 and 3, Fig. 3), as well as for the V/S method with different values of the amount of division of the time interval k, where $k = 100$ (Tables 4 and 5, Fig. 4) and $k = 300$ (Tables 6 and 7, Fig. 5).

Table 2. Hurst parameter calculation results for real traffic using R/S analysis

Hurst parameter value (traffic for 10.05.2017)	Hurst parameter value (traffic for 11.05.2017)	Traffic for Hurst parameter value (12.05.2017)
0,776487	0,755443	0,578647

Table 3. Hurst parameter calculation results for modeled traffic using R/S analysis

Function type	Hurst parameter value (10.05.2017)	Hurst parameter value (11.05.2017)	Hurst parameter value (12.05.2017)
sa(1, 1, t)	0,702865	0,757899	0,704181
sa(1/7, 3, t)	0,706573	0,762734	0,699697
sa(1/5, 1, t)	0,70686	0,759841	0,692659
sa(1/3, 1, t)	0,708239	0,749252	0,701112
sa(0.01, 0.1, t)	0,695051	0,745324	0,697232
sa(1, 7, t)	0,696894	0,755396	0,699143
sa(1, 1/3, t)	0,702687	0,762055	0,708672

In Figs. 3 and 4, the first column of the chart corresponds to the selected data of real traffic sample, the Hurst coefficient values of which are shown in Tables 3 and 4 in the first line. Comparing the results of calculations for real and modeled traffic shows a larger spread of computing results for real traffic samples compared with the model traffic samples. The comparison of the calculations by

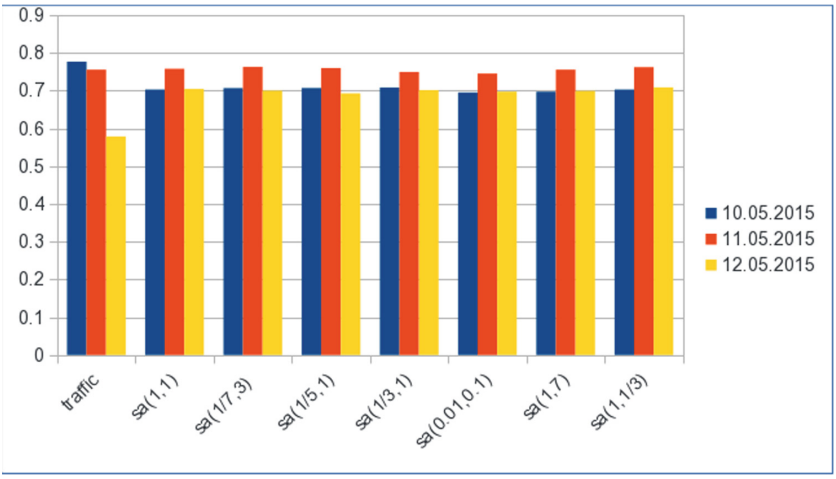


Fig. 3. Hurst parameter values for real and modeled traffic samples using R/S analysis.

Table 4. Hurst parameter calculation results for real traffic using V/S analysis for $k = 100$

Hurst parameter value (traffic for 10.05.2017)	Hurst parameter value (traffic for 11.05.2017)	Hurst parameter value (traffic for 12.05.2017)
0,472811	0,380671	0,313827

Table 5. Hurst parameter calculation results for modeled traffic using V/S analysis for $k = 100$

Function type	Hurst parameter value (10.05.2017)	Hurst parameter value (11.05.2017)	Hurst parameter value (12.05.2017)
$sa(1, 1, t)$	0,608668	0,605909	0,605868
$sa(1/7, 3, t)$	0,600347	0,595662	0,600275
$sa(1/5, 1, t)$	0,600398	0,596081	0,60033
$sa(1/3, 1, t)$	0,599482	0,596307	0,599413
$sa(0.01, 0.1, t)$	0,609914	0,607638	0,610486
$sa(1, 7, t)$	0,60995	0,60788	0,610553
$sa(1, 1/3, t)$	0,607961	0,607886	0,60801

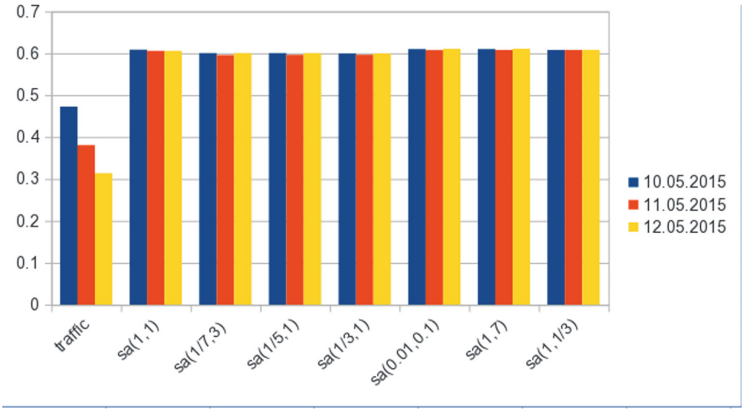


Fig. 4. Hurst parameter values for real and modeled traffic samples using V/S analysis for $k = 100$

Table 6. Hurst parameter calculation results for real traffic using V/S analysis for $k = 300$

Name	Hurst parameter value (10.05.2017)	Hurst parameter value (11.05.2017)	Hurst parameter value (12.05.2017)
Traffic	0,510714	0,442446	0,346464

Table 7. Hurst parameter calculation results for modeled traffic using V/S analysis for $k = 300$

Function type	Hurst parameter value (10.05.2017)	Hurst parameter value (11.05.2017)	Hurst parameter value (12.05.2017)
Traffic	0,510714	0,442446	0,346464
$sa(1, 1, t)$	0,642291	0,642654	0,64264
$sa(1/7, 3, t)$	0,636954	0,63623	0,636924
$sa(1/5, 1, t)$	0,637107	0,636658	0,637082
$sa(1/3, 1, t)$	0,636594	0,636614	0,636561
$sa(0.01, 0.1, t)$	0,642945	0,642956	0,642529
$sa(1, 7, t)$	0,643276	0,643293	0,642884
$sa(1, 1/3, t)$	0,642028	0,642069	0,642105

R/S method (Tables 1, 2 and 3, Fig. 3) and the V/S method (Tables 4, 5, 6 and 7, Figs. 4 and 5) shows the higher values of the calculated Hurst coefficient by about 0.7 according to the R/S method than by the V/S method is about 0.6.

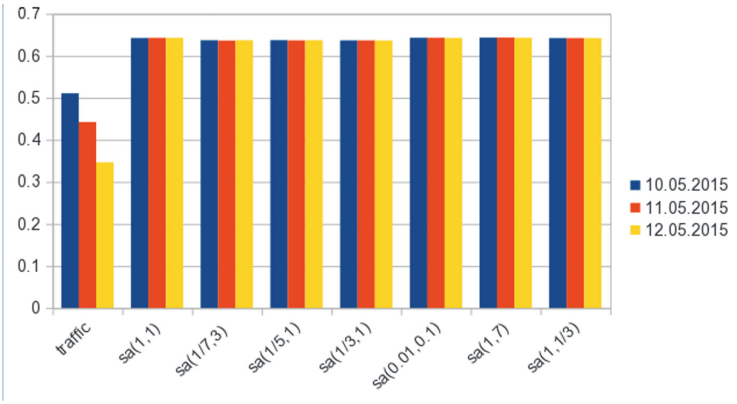


Fig. 5. Hurst parameter values for real and modeled traffic samples using V/S analysis for $k = 300$

Consequently, it can be concluded that for the studied traffic samples, the R/S method gives higher values of the Hurst coefficient.

The comparison of calculations of the Hurst coefficient for real traffic samples using the V/S method for different values of the coefficient k (Table 4, $k = 100$) and (Table 6, $k = 300$) shows that an increase of k value corresponds to increasing of the Hurst coefficient value. On the contrary, for modeled traffic samples, the increase of k value corresponds to reducing of the Hurst coefficient value (Table 5, $k = 100$) and (Table 7, $k = 300$).

6 Conclusions

New trends in the development of computer networks create the need for new approaches and strategies for their research, as well as a reappraisal of models designed to address such issues as scalability, elasticity, reliability, security, sustainability and application of traffic behavior patterns.

This paper shows improvements in the traffic behavior modeling method based on the consideration of traffic self-similarity parameters. Two methods for calculating the Hurst parameter were used: R/S plot method and V/S analysis method.

Committed studies have shown that for the traffic that is available in the network of the ACS NULP department, the method that gives the higher value of the Hurst parameter is the R/S method. This was observed in all cases that were considered. In addition, this method has a faster computation time compared to the V/S method in 7 times in average. So it can be concluded that the R/S method can be used for the implementation in the network nodes for determination of the Hurst parameter to improve the Ateb-modeling method, developed before.

The selected methods are implemented in the corresponding software. The work of the methods is illustrated in experimental calculations. The work has a

practical application for studying the self-similar properties of traffic. The growth of traffic volumes, transmitted through the network, as well as a significant increase in the proportion of video files in traffic, causes the use of additional traffic management tools directly at the nodes of the network. In order to manage traffic at a network node and redistribute it to reduce delays, it is efficient to use traffic modeling methods. Thereby, the investigation of self-similarity parameters of traffic helps to increase the effective traffic management in the nodes of the network.

References

1. The Zettabyte Era: Trends and Analysis. https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html
2. Gelembe, E.: Steps towards self-aware networks. Commun. ACM **52**(7), 66–75 (2009). https://doi.org/10.1109/CIT.2010.508
3. Toral-Cruz, H., Pathan, A.K., Ramirez Pacheco, J.C.: Accurate modeling of VoIP traffic QoS parameters in current and future networks with multifractal and Markov models. Math. Comput. Model. **57**(11–12), 2832–2845 (2013). https://doi.org/10.1016/j.mcm.2011.12.007
4. Dronyuk, I., Fedevych, O.: Traffic flows ateb-prediction method with fluctuation modeling using Dirac functions. Commun. Comput. Inf. Sci. **718**, 3–13 (2017). https://doi.org/10.1007/978-3-319-59767-6_1
5. Mishura, Y., Ral'chenko, K., Seleznev, O., Shevchenko, G.: Asymptotic properties of drift parameter estimator based on discrete observations of stochastic differential equation driven by fractional brownian motion. In: Korolyuk, V., Limnios, N., Mishura, Y., Sakhno, L., Shevchenko, G. (eds.) Modern Stochastics and Applications. SOIA, vol. 90, pp. 303–318. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-03512-3_17
6. Park, K., Willinger, W.: Self-similar network traffic: an overview. In: Self-Similar Network Traffic and Performance Evaluation. Wiley (2000). https://doi.org/10.1002/047120644X.ch1
7. Demydov, I., Klymash, M., Kharkhalis, Z., Strykhaliuk, B., Komada, P., Shedreyeva, I., Targeusizova, A., Iskakova, A.: The research of the availability at cloud service systems In: Proceedings Volume 10445, Photonics Applications in Astronomy, Communications, Industry, and High Energy Physics Experiments 2017, 104451V (2017). https://doi.org/10.1117/12.2280885
8. Gnatushenko, V.V., Ali, D.: Studies of self-similar processes traffic based on the ON/OFF model. In: Bulletin of the National University "Lviv Polytechnic", Series of "Computer Science and Information Technologies" - Lviv, No. 751, pp. 87–94 (2013). (in Ukrainian)
9. Ramirez-Pacheco, J.C., Torres-Roman, D., Toral-Cruz, H., Estrada-Vargas, L.: A high performance tool for the test of long-memory and self-similarity. In: Simulation Technologies in Networking and Communications: Selecting the Best Tool for the Test, pp. 93–114. CRC Press, Taylor & Francis Group, USA (2014). https://doi.org/10.1201/b17650-6
10. Estrada Vargas, L., Torres Romn, D., Toral-Cruz, H.: A study of wavelet analysis and data extraction from second-order self-similar time series. Math. Probl. Eng. **2013**, 1–14 (2013). https://doi.org/10.1155/2013/102834

11. Cajueiro, D., Tabak, B.: The rescaled variance statistic and the determination of the Hurst exponent. Math. Comput. Simul. **70**, 172–179 (2005). https://doi.org/10.1016/j.matcom.2005.06.005
12. Mandelbrot, B.B.: The Fractal Geometry of Nature. W. H. Freeman, NewYork (1982). https://doi.org/10.1119/1.13295. 550 p. 8
13. Senyk, P.: Reversal of incomplete Beta function. Ukrainian Math. J. **3**, 325–333 (1969). (in Ukrainian)

A Validation Method of a Real Network Device Model in the Riverbed Modeler Simulator

Dagmara Mazur(✉)

Faculty of Cybernetics, Military University of Technology,
ul. gen. Urbanowicza 2, 00-908 Warszawa, Poland
dagmara.mazur@wat.edu.pl

Abstract. The paper proposes a validation method of a model of a real routing device. In the developed method assessment of the device model accuracy is made with the use of statistical inference method. The paper presents results of research on the adequacy of the router model operation in the Riverbed Modeler simulator in relation to the operation of the real router in a real computer network environment. The router model was validated for the behaviour of selected queuing mechanisms. The obtained research results indicate the need to adapt the router model in the Riverbed Modeler simulator in the field of tested mechanisms before the model will be used to carry out research on these mechanisms on a large scale.

Keywords: Simulation · Validation · Riverbed Modeler
Queuing mechanisms

1 Introduction

The development of the Internet and web applications forces the evolution and appearance of new protocols and network mechanisms. Development or new technologies creation can be performed using real networks or simulators. Conducting research in both environments has its advantages and disadvantages.

Modifications of an existing protocol, implementations of a new protocol or a network mechanism in real network devices involves high costs. The construction of large real networks also means large financial and time outlays. However, studies conducted in such networks allow to get accurate and detailed results that may be observed in the target production environment.

By using simulators one can create only the model of real devices. Such a model is only an approximate playback of phenomena or behaviours of a given device. Computer simulation refers to mapping the actual behaviour of the device through a computer program [1]. Simulators allow only to conduct research in a virtual and not real environment. On the other hand, simulators have many advantages [2] and some of them are as follows: possibility to study a very complex phenomena, relatively easy way to change device parameters, a significant

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 26–39, 2018.
https://doi.org/10.1007/978-3-319-92459-5_3

reduction of financial outlays needed to carry out the research and short time required for reconfiguration of all devices. There are many simulators available on the market, some examples are as follows: ns-3 [3], Riverbed Modeler [4], OMNeT++ [5], REAL [6], NetSim [7], QualNet [8], J-Sim [9], and SSFNet [10].

The aim of this paper is to present a developed method of checking whether a given real network device model properly reflects this device in the scope of a tested application. The developed method allows to answer the following questions: whether the results of tests performed in the simulator coincide with the results which were obtained in the real environment; whether a given model of the device can be used for tests carried out on a large scale, without the use of real devices and a computer network.

2 Related Works

In the literature one can find results of tests carried out using various simulators confronted with results obtained in real networks. The paper [11] focuses on the comparison of results obtained in the OPNET Modeler simulator (OPNET Modeler is the previous version of the Riverbed Modeler simulator), the ns-2 simulator, and the real network. The research concerns the network transmission study for two types of data streams: the Constant Bit Rate (CBR) and the File Transfer Protocol (FTP). The another paper [12] also concerns a transmission of the CBR data stream, but the FTP data stream is not taken into account. Moreover, the authors compared results obtained in simulators considered in the previous publication with results obtained in the QualNet simulator. The paper [13] presents results of research on packets queuing mechanisms in the ns-2 simulator and compares them with results of testing the same mechanisms in a real network. Respectively, work described in the paper [14] focuses on model credibility verification for network devices used for military purposes in the OPNET Modeler simulator. The research concerned packets queuing mechanisms operation in terms of the performance.

This paper presents research results on the adequacy of queuing mechanisms operation implemented in the router model in the Riverbed Modeler simulator. Simulation results are compared to research results achieved during testing the operation of these mechanisms in a real network. The Riverbed Modeler simulator was chosen for this research due to its popularity in academic, commercial and industrial environments. The paper expands work presented in [13] and [14] with different set of selected queuing mechanisms, and focuses on the comparison of mechanisms behaviour. Additionally, in the developed method assessment, the accuracy of the device model is performed with the use of statistical inference method.

3 The Validation Method

The validation [15] is a process in which it is assessed whether a model intended to represent any real phenomenon in the expected manner reflects this phenomenon in the field of the tested application.

Figure 1 shows a scheme of the validation process of the real device model. The first step in the validation process is to conduct independent research in a real and simulated environment. In the next step obtained research results are compared with each other. If the developed device model does not reflect the real phenomenon in the expected way, attributes of the device model should be adjusted, and the research should be repeated in the simulated environment. Then, the comparison of test results should be repeated. The device model attributes are iteratively adjusted until the developed device model is reflecting the real phenomenon in the expected way.

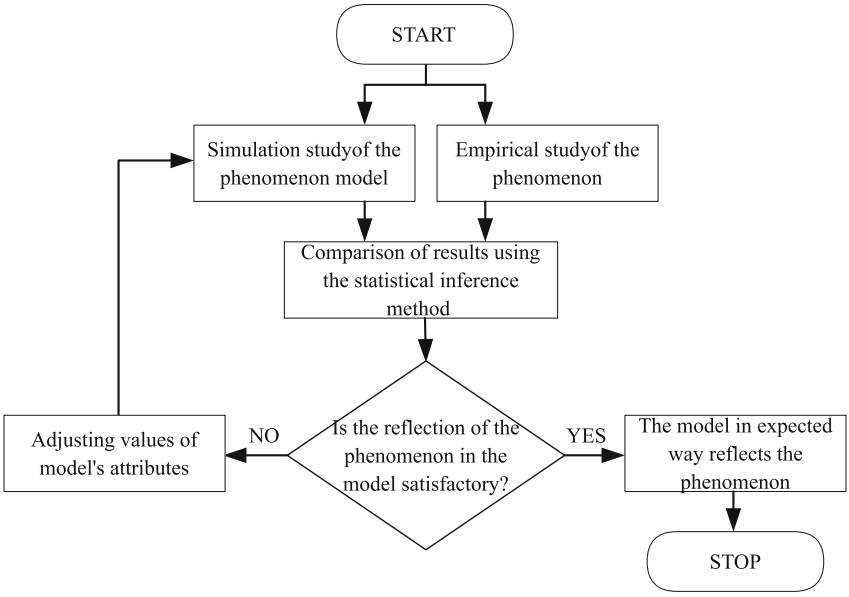


Fig. 1. Scheme of the developed validation process of the real device model.

Assessment of the accuracy of the reflection of the real phenomenon in the device model can be based on selected, measurable technical parameters. These parameters are characteristic for the researched phenomenon. The evaluation accuracy of the device model may focuses on the behaviour study of the device model (characteristics of a given phenomenon) or quantitative study of these behaviours (values of given technical parameters).

The statistical inference method should be used for the device model evaluation when quantitative study is subject to the research [16]. The method itself allows to determine whether there is a significant statistical difference between data samples received in the real and simulated environment. The method consists of the following stages:

STAGE I: Formulating the null and alternative hypotheses, H_0 and H_1 respectively.

The null hypothesis H_0 is the one to be checked. The alternative hypothesis H_1 is the one to be accepted when the null hypothesis is rejected.

STAGE II: Setting a level of significance α.

The level of significance α means probability of making mistake which consists in rejecting the hypothesis H_0 despite H_0 is true.

STAGE III: Selecting a statistical test to check the null hypothesis H_0.

The statistical test selection depends on data samples type.

STAGE IV: Calculation of the value T of the selected statistical test based on data samples.

STAGE V: Finding a critical value t for the fixed level of significance α.

The critical value t comes from statistical tables. Based on value t a decision is made about acceptance or rejection of the null hypothesis H_0.

STAGE VI: Making decision about acceptance or rejection of the null hypothesis H_0 at the given level of significance α.

The null hypothesis H_0 should be accepted when $|T| < t$. However when the condition $|T| < t$ is not met and the condition $|T| \geq t$ is true then the null hypothesis H_0 should be rejected and the alternative hypothesis H_1 should be accepted.

The acceptance of the null hypothesis ends the validation process with a positive result. The positive result means that the existing approximations of reality in the device model do not significantly affect the mapped real phenomenon. Moreover, the statistical method application proves positive result of the validation to be objective. The validated simulator model can be used to carry out the same research in the case of a large scale network.

4 Completed Research

4.1 Research Environment

Research consist in checking the adequacy of operation of the router model in the Riverbed Modeler simulator in relation to the operation of the real device. The router model was validated for the behaviour of selected queuing mechanisms. To conduct these research two environments should be prepared: real and simulated ones. The physical topology of the real network is shown in the Fig. 2, and the topology of the simulated network is shown in Fig. 3.

The real computer network has been equipped with two Cisco 2620 routers: R1 and R2, two switches: S1 and S2 and traffic generator TG and traffic receiver TR. The generator and the traffic receiver were realized using application IP Traffic - Test and Measure [17]. In order to obtain correct research results traffic generator TG and traffic receiver TR must indicate simultaneously the same system time. Therefore, their operating system clocks were synchronized with the clock of the external NTP server [18]. Both switches were connected with other network elements using ethernet link with the bandwidth of 100 Mb/s, and both routers were connected using serial link with the bandwidth of 128 kb/s. This allowed the creation of a so-called bottleneck on the serial link, which is necessary to observe the characteristic operation of selected queuing mechanisms.

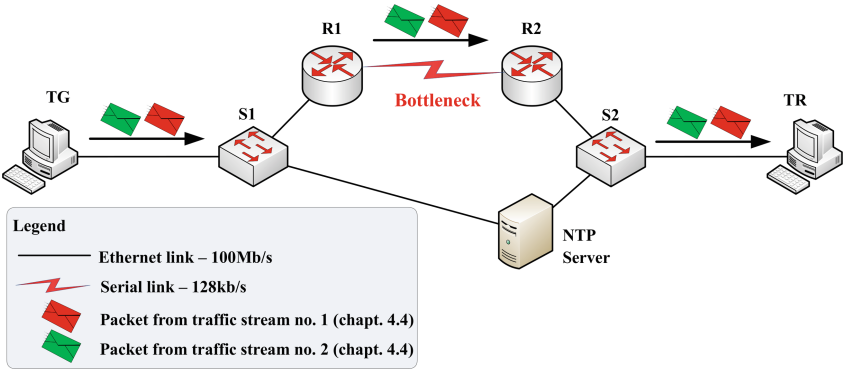


Fig. 2. Physical topology of the real network.

Analogically to the real computer network, the computer network in the Riverbed Modeler simulator was built. The simulator network consists of two Cisco 2620 routers: R_1 and R_2, two switches: S_1 and S_2, traffic generators and traffic receivers (ethernet_ip_workstation_adv [19]): TG_1, TG_2, TR_1, and TR_2. There are more traffic generators and receivers than in the case of the real network. This is because the Riverbed Modeler simulator traffic generator can generate only one type of traffic – one traffic stream. In the simulated network, traffic generators and receivers are not connected to the NTP server, because their clocks are synchronized by the simulator. As in the real network, both switches were connected to other network elements using ethernet link with the bandwidth of 100 Mb/s, and both routers were connected using serial link with the bandwidth of 128 kb/s.

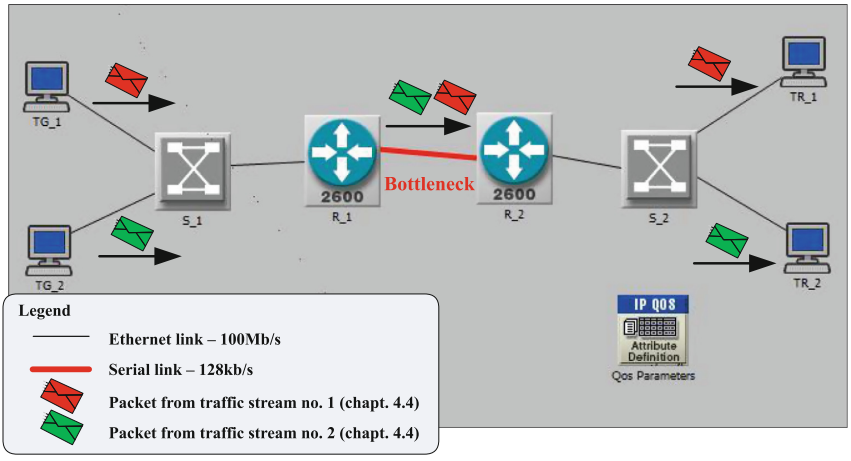


Fig. 3. Physical topology of the simulated network.

4.2 Phenomena and Metrics Selected for the Validation of the Router Model

In order to validate the Cisco 2620 router model in the Riverbed Modeler simulator, there were selected following mechanisms and phenomena:

1. Priority Queuing (PQ) [20]: appropriation of the whole bandwidth by the highest priority traffic.
2. Custom Queuing (CQ) [20]: preservation of the allocated bandwidth to a given queue and rejection of packets that exceed the bandwidth allocated to the given queue when the interface is saturated.
3. Low Latency Queuing (LLQ) [20]: limitation of the bandwidth on the priority queue and rejection of packets that exceed the bandwidth allocated to the given queue when the interface is saturated.

The accuracy of the reflection of the above mechanisms in the validated router model was evaluated on the basis of the throughput parameter.

4.3 Routers Configuration

As part of the router model validation three tests: A, B, C were performed. The routers configuration during each test is described below:

1. Test A: PQ mechanism was configured on router R1 and R_1.
2. Test B: CQ mechanism was configured on router R1 and R_1; In this mechanism two queues were created; The first queue was intended for high-priority traffic operation and was allocated 2/3 of the available bandwidth to it; The second queue was dedicated to low-priority traffic and was allocated 1/3 of the available bandwidth to it.
3. Test C: LLQ mechanism was configured on router R1 and R_1; In this mechanism two queues were created; The first queue is a priority queue, which was intended for high-priority traffic operation and was allocated 64kb/s of available bandwidth to it; The second queue was intended for low-priority traffic and it was no restrictions imposed on it on the allocation of available bandwidth.

4.4 Traffic Generators Configuration

Two traffic streams of constant bit rate (CBR) were generated during each test (A, B and C). It means that the Packet Size and the Inter Arrival Time parameters were configured for both generated traffic stream. Each test was executed in the real network and in the Riverbed Modeler simulator. Table 1 presents detailed settings for the parameters of both traffic streams, and Fig. 4 shows the time ranges in which traffic flows were generated.

Additionally in both traffic stream generators the MTU size parameter was set the same. The MTU size parameter is a crucial factor which could influence on a traffic streams characteristic. The generators ability to generate expected

Table 1. Traffic streams parameters.

Number of traffic stream	Throughput at layer 2 level of the ISO/OSI model	Throughput at layer 4–7 level of the ISO/OSI model	Priority of traffic stream [20]
Traffic stream no. 1	100 kb/s	89,2 kb/s	High (ToS = 6)
Traffic stream no. 2	80 kb/s	69,2 kb/s	Medium (ToS = 4)

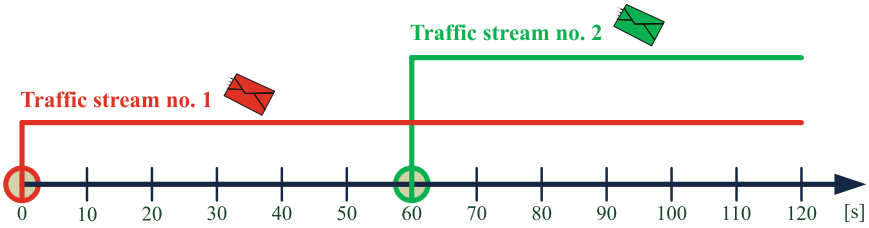


Fig. 4. Time intervals of generating traffic streams.

traffic streams characteristic was automatically validated by observation of the traffic stream no 1 in the first 60 s of each test.

It should be noticed that the total sum of the throughput of both traffic streams exceeds the available bandwidth of the serial link. Such throughput values of traffic streams allow to observe the characteristic operation of each tested queuing mechanism. Higher throughput values of traffic streams would also give this opportunity.

4.5 Research Results

All performed tests A, B, and C were carried out in the real network and in the Riverbed Modeler simulator. After the first series of tests, it turned out that the throughput values of traffic streams sent by router R_1 in the simulator are about 25% smaller than the values received on the router R1 in the real network. The study of this case revealed the fact that both routers (R1 and R_1) with configured the queuing mechanism on its interface, automatically reserve only 75% of the interface bandwidth for the needs of defined queues. In turn the real router was using the remaining 25% of the interface bandwidth when no traffic exist outside defined queues. The observed behaviour is the first discrepancy found in the implementation of the router model in relation to the real device.

After customizing the configuration of router model in the Riverbed Modeler simulator all tests were repeated in the real and simulated network. Adjusting the configuration of the router model depends on setting the value of 100 in the field max-reserved-bandwidth on its serial interface [21].

Data samples received from the real and simulated environment were checked using the statistical inference method described in Sect. 3. Tests A, B and C were in the scope of the statistical verification. One can find below the procedure used for each stage of the mentioned method.

STAGE I: The following research hypotheses were formulated:

The null hypothesis H_0 – The average of traffic stream throughput in the real environment is not significantly statistically different from the average throughput obtained in the Riverbed Modeler simulator.

The alternative hypothesis H_1 – The average of traffic stream throughput in the real environment is significantly statistically different from the average throughput obtained in the Riverbed Modeler simulator.

STAGE II: Assumed $\alpha = 0,05$ as a level of significance.

STAGE III: Statistical Student's t-test was selected, because all obtained results have a normal distribution, sets of traffic stream throughput results in the real environment and in the Riverbed Modeler simulator have similar numbers, variances of results obtained in both environments are similar – homogeneous and this results are measured on interval scale (interval) – samples can be ordered and have a unit of measure [kb/s].

STAGE IV: A statistical Student's t-test T value calculation with the following equation:

$$T = \frac{\overline{X_1} - \overline{X_2}}{S_{x_1 - x_2}} \tag{1}$$

$$S_{x_1 - x_2} = \sqrt{\frac{(n_1 - 1) \cdot s_1^2 + (n_2 - 1) \cdot s_2^2}{n_1 + n_2 - 2} \cdot \left(\frac{1}{n_1} + \frac{1}{n_2}\right)} \tag{2}$$

where:

$\overline{X_1}$ – the average value of the traffic stream throughput in the real environment

$\overline{X_2}$ – the average value of the traffic stream throughput in the Riverbed Modeler simulator

s_1^2 – the variance of traffic stream throughput in a real environment

s_2^2 – the variance of traffic stream throughput in the Riverbed Modeler simulator

n_1 – the samples number of traffic stream throughput in a real environment

n_2 – the samples number of traffic stream throughput in a real Riverbed Modeler simulator

STAGE V: The critical value t is read from the Student's t-distribution tables for the level of significance $\alpha = 0,05$ and the number of degrees k of freedom expressed by the formula:

$$k = n_1 + n_2 - 2 \tag{3}$$

STAGE VI: When the following condition is met:

$$|T| < t \tag{4}$$

Acceptance decision on the null hypothesis H_0 is making and the alternative hypothesis is rejecting H_1. Otherwise, the alternative hypothesis H_1 is accepted.

Expected Behaviours of Configured Queuing Mechanisms. Figures 5, 6 and 7 show respectively operating schemes of configured PQ, CQ, and LLQ mechanisms in the real and simulated network. Each queuing mechanism is implemented on routers R1 and R_1 for respective test execution (A, B, and C). Routers R1 and R_1 receive packets on input interface from traffic streams no. 1 and no. 2. Packets from each traffic stream are classified based on the assigned priority (ToS field values) and placed in the appropriate queue. Below describes how each queuing mechanism continues to work according to the configuration described in Sect. 4.3.

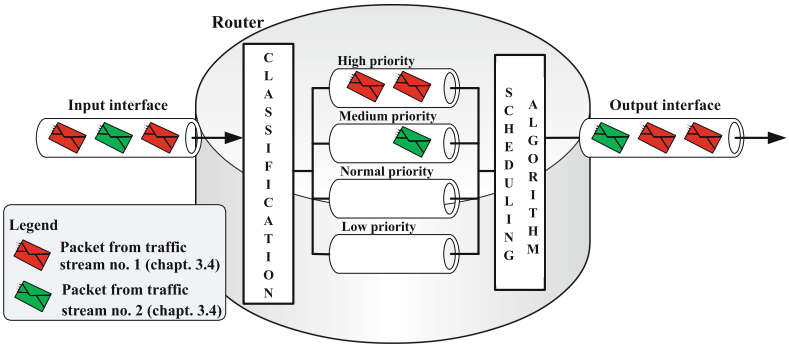


Fig. 5. The scheme of operation of the configured PQ mechanism.

PQ mechanism (Figure 5): Packets from traffic stream no. 1 are routed to the high priority queue, and packets from traffic stream no. 2 are routed to the medium priority queue; then the scheduling algorithm is putting the packets in the router's interface as per the rule: packets from a high priority queue have priority over packets from other queues.

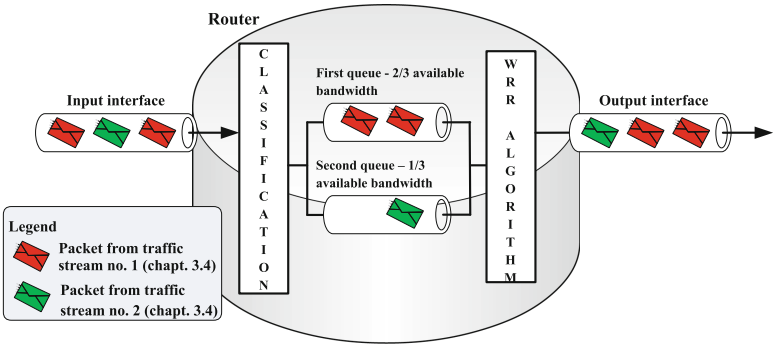


Fig. 6. The scheme of operation of the configured CQ mechanism.

CQ mechanism (Figure 6): Packets from traffic stream no. 1 are routed to the first queue, and packets from traffic stream no. 2 are routed to the second queue; then the Weighted Round Robin algorithm (WRR) is putting packets in the router's output interface. Packets from the first queue should be sent as first until their total number of bits reach the value set for this queue (2/3 of the serial link bandwidth).

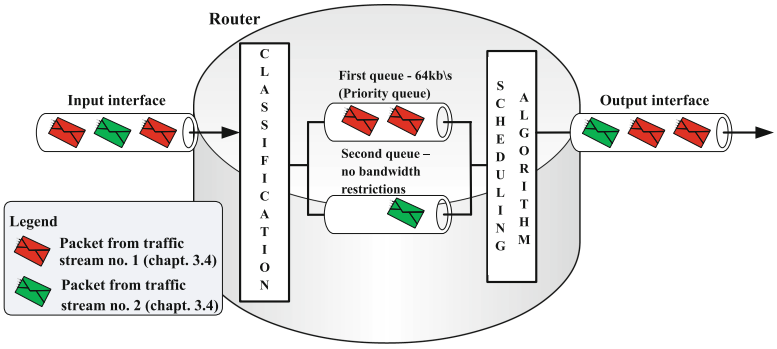


Fig. 7. The scheme of operation of the configured LLQ mechanism.

LLQ mechanism (Figure 7): Packets from traffic stream no. 1 are routed to the first queue – priority queue, and packets from traffic stream no. 2 are routed to the second queue. Then the packets are scheduled in the router's output interface. Packets from the first queue should be sent as first until their total number of bits reach the value set for this queue (64 kb/s – at the layer 2 level of the ISO/OSI model).

Test A – Research Results. Figure 8 shows results of the conducted test A. Research results shown that the throughput of the traffic stream no. 1 on router R_1 is less by about 4kb/s than the throughput of the traffic stream no. 1 on router R1. Mentioned difference in the throughput is observed from the moment when the traffic stream no. 2 appears on the input interface of the R_1 and R1 routers. As a consequence, the described phenomenon causes the traffic stream no. 2 in the Riverbed Modeler simulator to be about 4 kb/s higher than in the real network. This means that the traffic stream no. 1 in the Riverbed Modeler simulator does not cover the whole bandwidth.

The source of observed decreases and increases of the throughput on the router's output interface in the Riverbed Modeler simulator may be the automatic application of mechanisms to optimize packet traffic efficiency at the crowded router interface. Examples of such mechanisms include: compression of packet headers, packet fragmentation or frame size modification at the layer 2 level of the ISO/OSI model. During the research both simulated and real routers were not modified with mentioned mechanisms.

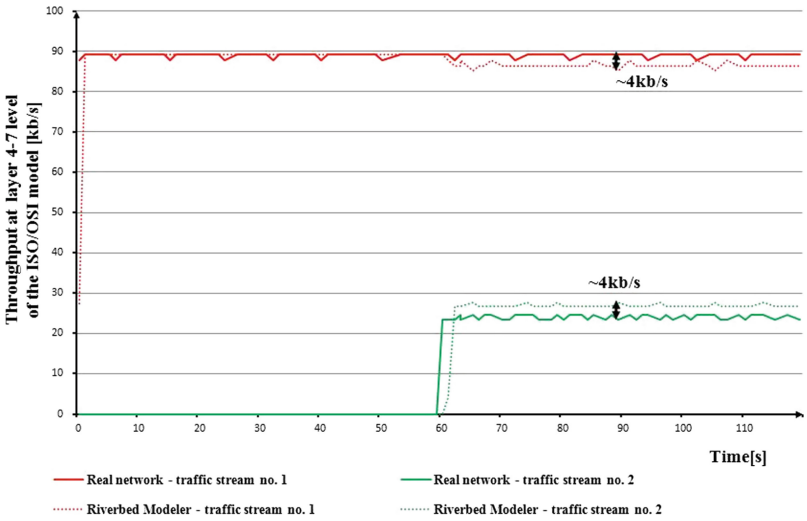


Fig. 8. Graph of the throughput dependence on the time after conducting the test A.

After usage the statistical inference method, it turned out that for both traffic streams the alternative hypothesis was accepted. The results of the average throughput of the traffic stream no. 1 and no. 2 in the real environment significantly differ statistically from the results of the average throughput of the traffic stream no. 1 and no. 2 obtained in the simulator. This means a negative result of the validation of the Cisco 2620 router model in the Riverbed Modeler simulator. Current implementation of the router model cannot be used for conducting the research of the PQ mechanism on a large scale network.

A detailed analysis of the code of the implemented PQ mechanism in the Riverbed Modeler simulator and its impact on the obtained research results will be the subject of further research.

Test B – Research Results. Figure 9 shows the results of the conducted test B. The research results shown that the Cisco 2620 router model in the Riverbed Modeler simulator perfectly reflects the behaviour of the real router for the CQ mechanism. After 60 s of test B duration, the throughput of the traffic stream no. 1 is equal to the bandwidth allocated to the first queue (approximately 77 kb/s – 2/3 of the available bandwidth), and the throughput of the traffic stream no. 2 is equal to the bandwidth allocated to the second queue (approximately 37 kb/s – 2/3 of available bandwidth). The rest of packets from both streams is discarded.

As a result of usage the statistical inference method, it turned out that for both traffic streams the zero hypothesis was accepted. The results of the average throughput of the traffic stream no. 1 and no. 2 in the real environment do not significantly differ statistically from the results of the average throughput

of the traffic stream no. 1 and no. 2 obtained in the simulator. This means a positive result of the validation of the Cisco 2620 router model in the Riverbed Modeler simulator. Current implementation of the router model can be used for conducting the research of the CQ mechanism on a large scale network.

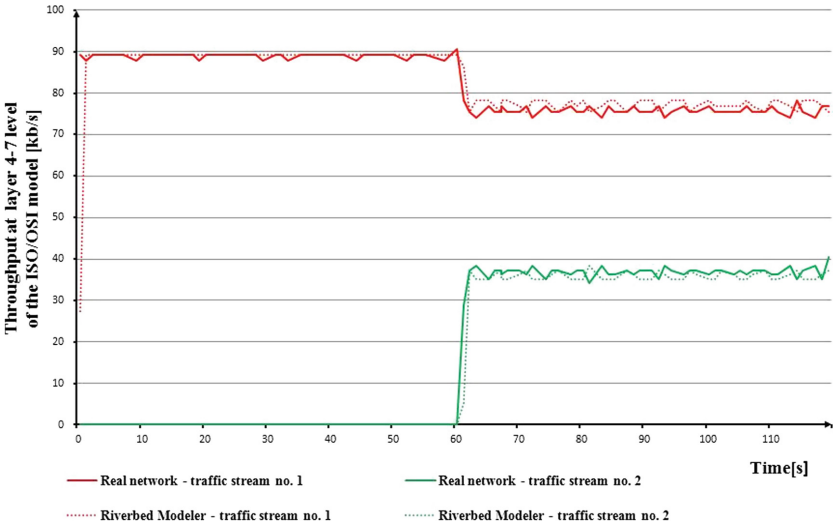


Fig. 9. Graph of the throughput dependence on the time after conducting the test B.

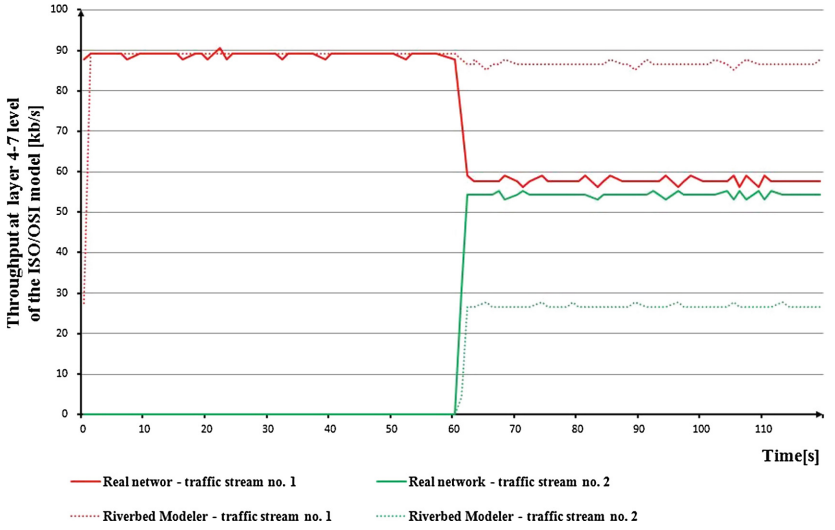


Fig. 10. Graph of the throughput dependence on the time after conducting the test C.

Test C – Research Results. Figure 10 shows the results of the conducted test C. The research results shown that the Cisco 2620 router model in the Riverbed Modeler simulator does not introduce a bandwidth limitation for the first queue (priority queue). Therefore, the research results conducted for the LLQ and PQ mechanisms in the simulated network overlap with each other for both traffic streams.

Despite such divergent research results, procedures of the statistical inference method was conducted. The result of the Cisco 2620 router model validation in the Riverbed Modeler simulator is negative as the null hypothesis was rejected. Current implementation of the router model cannot be used for conducting the research of the LLQ mechanism on a large scale network.

5 Summary

The paper describes the validation method of the real device model. In the developed method assessment of the accuracy of the device model is made with use of the statistical inference method.

The use of the developed method was demonstrated on the example of validation of the Cisco 2620 router model. The paper presents research results on the adequacy of operation of the router model in the Riverbed Modeler simulator in relation to the operation of the real router in the real computer network. The router model was validated for the behaviour of the PQ, CQ and LLQ mechanisms. Accuracy assessment of the router model was made on the basis of the throughput parameter.

After the first series of tests, it turned out that throughput values of traffic streams received in the simulator were about 25% smaller than values received in the real network. The developed model did not reflect the real phenomenon in the expected way, so the configuration of the device model in the simulator was adapted in accordance with the scheme of the validation process shown in Fig. 1. Then the accuracy research of the model was repeated. For this purpose, the statistical inference method was used, in which Student's t-test was selected to verify the research hypothesis.

After another iteration of the conducted research, the validation of the Cisco 2620 router model received a positive result only for the CQ mechanism. For the PQ and LLQ mechanisms, the results of the average throughput of the traffic stream no. 1 and no. 2 in the real environment significantly differ statistically from the results of the average throughput of the traffic stream no. 1 and no. 2 obtained in the simulator. This involves a negative result of the validation of the Cisco 2620 router model in the Riverbed Modeler simulator. The current implementation of the router model cannot be used for conducting a research of the PQ and LLQ mechanisms on a large scale network. Thus, another validation of the router model in the field of the PQ and LLQ mechanisms should be preceded by a modification of the implementation of these mechanisms in the simulator; and that will be the subject of the further research.

References

1. Balci, O.: Validation, verification, and testing techniques throughout the life cycle of a simulation study. In: IEEE Winter Simulation Conference, pp. 215–220 (1994)
2. Breslau, L., Estrin, D., Fall, K., Floyd, S., Heidemann, J., Helmy, A., Huang, P., McCanne, S., Varadhan, K., Xu, Y., Yu, H.: Advances in network simulation. IEEE Comput. Magaz. **33**, 59–67 (2000)
3. NS, Discrete Event Network Simulator. http://www.nsnam.org. Accessed 21 Feb 2018
4. Riverbed Modeller, Network Simulator. https://www.riverbed.com/. Accessed 21 Feb 2018
5. OMNeT++, Discrete Event Simulator. https://www.omnetpp.org/. Accessed 21 Feb 2018
6. REAL, Network Simulator. http://www.cs.cornell.edu/skeshav/real/overview.html. Accessed 21 Feb 2018
7. NetSim, Network Simulation and Emulation Platform. http://www.tetcos.com/index.html. Accessed 21 Feb 2018
8. QualNet, Network Simulator. www.scalable-networks.com. Accessed 21 Feb 2018
9. J-Sim, Java-based simulation system. http://www.physiome.org/jsim/. Accessed 21 Feb 2018
10. SSFNet. http://www.ssfnet.org/. Accessed 21 Feb 2018
11. Lucio, G.F., Paredes-Farrera, M., Jammeh, E., Fleury, M., Reed, M.J.: OPNET modeler and NS-2: comparing the accuracy of network simulators for packet level analysis using a network Testbed. In: ICOSMO, vol. 2, pp. 700–707 (2003)
12. Puneet R., Srinath P., Raghuraman R.: Bridging the gap between the reality and simulations: an Ethernet case study. In: IEEE 9th International Conference on Information Technology (ICIT 2006), pp. 52–55 (2006)
13. Andreozzi, S.: Differentiated services: an experimental vs. simulated case study. In: IEEE Seventh International Symposium on Computers and Communications, pp. 383–390 (2002)
14. Boltjes, B., Thiele, F., Fernandez Diaz, I.: Credibility and validation of simulation models for tactical IP networks. In: IEEE Military Communications Conference, pp. 1–10 (2006)
15. Heidemann, J.: Expanding confidence in network simulations. IEEE Netw. Magaz. **15**, 58–63 (2001)
16. Kowalski, L.: Statystyka. Bel Studio, Warsaw (2005)
17. IP Traffic - Test and Measure. https://www.zticommunications.com/iptraffic/. Accessed 21 Feb 2018
18. RFC 958: Network Time Protocol (NTP). http://tools.ietf.org/html/rfc958. Accessed 21 Feb 2018
19. Adarshpal, S.S., Vasil, Y.H.: The Practical Riverbed®User Guide for Computer Network Simulation. CRC Press, Boca Raton (2013)
20. Odom, W., Cavanaugh, M.: Cisco QOS Exam Certification Guide. Cisco Press, Indianapolis (2004)
21. Cisco website. http://www.cisco.com/c/en/us/support/docs/quality-of-service-qos/qos-packet-marking/10100-priorityvsbw.html. Accessed 21 Feb 2018

Energy Efficient MPTCP Transmission Over Channels with a Common Bottleneck

Michał Morawski(✉) and Przemysław Ignaciuk

Institute of Information Technology, Lodz University of Technology,
215 Wólczańska St., 90-924 Łódź, Poland
{michal.morawski,przemyslaw.ignaciuk}@p.lodz.pl

Abstract. The majority of networking devices currently available are equipped with a few communication interfaces. However, the design of the fundamental transport protocol – TCP – prohibits using them simultaneously. Recently, a variant of TCP – Multipath TCP (MPTCP) – that allows for parallel transmission over different paths has been developed. The protocol itself does not define any particular way of splitting the application data stream. Taking into account the proliferation of mobile terminals, a good quality indicator is the amount of energy dissipated by the communicating device. The previous works in the field assume that the paths established by the MPTCP stack are independent. This paper discusses the theoretical and practical aspects of stream splitting when the paths influence each other through a common bottleneck. A comparison with the reference solution is provided. The tests conducted in the public networking environment show a substantial improvement in energy economy.

Keywords: Multipath TCP · Load balancing · Energy efficiency

1 Introduction

Nowadays, the networking devices, like phones, laptops, or servers, have installed multiple communication interfaces (NICs), e.g., Ethernet, WiFi, or cellular. However, the design restrictions of the protocols created in the past permit only one interface to be actively engaged in serving the application data stream at a time. The other interfaces are incorporated if the active one fails, or if the application dictates a switch according to certain quality measure. Such behavior is no longer satisfactory owing to the widespread use of radio technologies. Trying to maintain the ongoing session within a chosen radio channel, with highly variable temporal characteristics (determined by the distance from the access point, nearby obstacles, node mobility, interferences, etc.), has negative impact on the user's experience [1,2]. Instead, one would opt for a communication solution in which the traffic is dynamically distributed among the available interfaces, with explicit consideration of their current transfer capabilities.

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 40–51, 2018.
https://doi.org/10.1007/978-3-319-92459-5_4

The recent proposals, like Multipath TCP (MPTCP) [3–5] promise to ensure truly parallel and load-balanced traffic distribution. However, the reference implementation of MPTCP [6] covers only the general protocol issues, leaving area for further improvements and adjustments. One of such unresolved challenges – addressed in this work – is formulating a method of MPTCP stream splitting into multiple sub-streams so that selected performance criteria are satisfied. As the popularity and functional spectrum of mobile computers have outgrown the progress in battery design, reducing the energy expenditure becomes a critical factor to consider within the ever more stringent service and environmental constraints (green networking) [7]. Therefore, in the presented approach, the emphasis is placed on improving the end user's experience through better energy economy while handling the transmission at the communicating terminals.

At a mobile device, the amount of dissipated energy depends both on the power efficiency of NICs and application level activity, i.e., turning on and off the display, triggering additional cores, GPU, or external sensors. For instance, one can observe faster battery depletion when the data is sent though a secure connection rather than in a plain-text form. The power drain rate also increases when the information regarding the transferred data is rendered on the screen. Therefore, in order to reduce the energy expenditure while effectuating the network connectivity, one should shorten the time of data transfer and allow the user to switch off the screen (and additional hardware components), or put the application in an idle state. This paper proposes a new load-balancing solution for the MPTCP scheduler, particularly well-suited for battery-powered devices equipped with a few independent interfaces. Although addressing similar problems as in the current literature [8–11], the idea presented here differs both in the objectives and in the way the traffic is distributed among the interfaces. The paper focuses on the situation when the sub-streams directed through different logical interfaces influence each other before reaching the destination, The transmission through independent channels is considered as a special case, only. The way the MPTCP stream is divided into sub-streams by the scheduler is determined as a solution of optimization problem set in a mathematical framework of power dissipation. The performance assessment and comparison with the reference scheduler is not restricted to simulations but involves public networks and real networking devices.

2 Multipath Data Transfer

The system architecture is illustrated in Fig. 1. In the considered communication scenario, the user application seeks to send B bits of data to a remote peer. The data is placed in the buffer held by a local MPTCP controller. The MPTCP controller opens sub-channels and distributes the traffic among the corresponding sub-streams that are left under the control of the ordinary, single path TCP (SPTCP). The SPTCP flow control is executed separately for each sub-channel. The research objective is to design an efficient load-balancing algorithm for the MPTCP traffic scheduler. The operation of SPTCP within the sub-channels, e.g., CUBIC, is not affected.

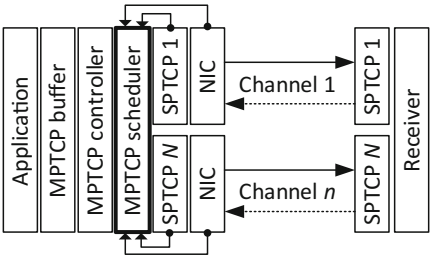


Fig. 1. MPTCP architecture with local feedback – the current power profile of NIC and SPTCP internals. Solid line – data, dashed line – acknowledgments.

The specification of MPTCP does not define how the user's data should be distributed among the sub-channels. Sometimes, it is appropriate to use only one path and switch to another only when the quality of service drops below a predefined level. The primary intention of MPTCP, however, is to employ multiple channels simultaneously. In its reference implementation [6], three schedulers have been defined:

(a) round-robin – evenly distribute subsequent segments among the sub-channels (insignificant in practice, serve as a theoretical baseline),
(b) redundant – send the same segment through all the available sub-channels – results in low latency but leads to increased bandwidth consumption,
(c) default – send the data using the channel with the lowest smoothed round-trip time (SRTT). Smoothing is performed by the filter typical for the selected SPTCP dialect.

Solution (c) – proposed to increase the total effective throughput – is strongly recommended by the authors of implementation [6]. In this paper, this solution is treated as a reference one. Other schedulers described in the current literature, e.g., [8–11], have not been adopted in the formal development works. On the other hand, the default scheduler suffers from a number of disadvantages:

– *sticky* behavior [12] – when the smoothed round-trip time (SRTT) differs a little among the active channels, the one with the shortest SRTT value is used exclusively,
– imprecise SRTT estimates – and thus wrong channel selection – originating from the buffer bloat (different, but generally large, queue lengths in the buffers of intermediate devices on the established paths),
– Karn's algorithm implications,
– spurious acknowledgments – if the channels differ much in their properties, then the retransmission timeout (RTO) at the MPTCP level can be shorter than the variance of SRTT in the 'worse' ones; then, the MPTCP stack mistakenly signals the sender that a segment has been lost;
– head-of-line blocking – if the buffers are too short at the end-points, then some data cannot be temporary sent via the available channel because there are unacknowledged data transmitted previously through the 'worse' ones.

3 Energy Considerations and Optimal Load Distribution

In the mobile devices, the GPU, CPU, and display tend to consume more energy than the network interfaces. The NIC energy expenditure depends on the transmission power, noise level, number of retransmissions, and control plane activity. These factors differ among the interfaces and their mode of operation. In general, the transmit power is highly variable in time [13], but even the median measurements differ much. The power effectiveness for the transmission can be as low as 2 [mW/Mbps] for cable NIC (e.g., Ethernet), and as high as 440 [mW/Mbps] for LTE NIC. Similarly, the power necessary to keep the interface in the operational state can be as low as 90 [mW], and as high as 1440 [mW], for the same interfaces, respectively [10]. Hence, the energy dissipated by the NICs does not have to be comparable with the energy consumed by the operating system and application activities (200–300 mW in low-end systems up to 300 W in high-end ones). Therefore, in order to reduce the overall energy expenditure during the data transfer the following optimization problem may be considered – minimize ϑ given by

$$\vartheta = \sum_{i=1}^{n} \vartheta_i + D \max T_i \tag{1}$$

subject to $T_i > 0$, where

$$\vartheta_i = \int_0^{T_i} p_i(t)dt \tag{2}$$

is the total energy dissipated by interface $i \in [1, n]$, n being the number of interfaces, $p_i(t) > 0$ is the power necessary to transmit the data through this interface at time t, T_i is the total transmission time at this interface, and $D > 0$ is the power consumed by non-NIC system components (display, additional cores, sensors, etc.). D can be obtained as a difference between the power consumed at a given moment and the power drained when the system is in an idle state. Meanwhile, the transmit power $p_i(t)$ [W] is a function of the power efficiency of NIC – $\rho_i(t)$ [W/bit], the number of bits sent through channel i – b_i, and the power necessary to keep the NIC in the operational state – w_i [W]. It can be expressed as

$$p_i(t) = \rho_i(t)b_i + w_i(t), \tag{3}$$

with $\rho_i(t)$ and $w_i(t)$ being functions of:

- the distance from the end-point to the hub (access point, head-end, or BTS),
- the mode of operation, which includes the selection standard 'b', 'g', 'n', 'ac', band, RTS, or CCA mode in WiFi and band, GPRS, or the revision of the UMTS class in the case of cellular interfaces,
- interferences level, number of retransmissions at the link layer.

Due to large number of factors affecting $\rho_i(t)$ and $w_i(t)$, these profiles should be directly measured by the communication end-point (e.g., such approach is suggested in [8]). The profile varies with time as a result of node mobility, controller activity, or other terminals behavior. Here, piece-wise constant evolution is assumed.

4 Analytical Background

Often, the paths established by the MPTCP subsystem are independent, i.e., there is no common bottleneck between the communicating peers. An energy-oriented solution for the independent channels case has been presented in [13]. However, such premise is not always justified. The streams may encounter a common bottleneck, or interfere through a third-party connection. The mutual dependency may be persistent or temporary, and may result in network or application performance decrease and is both difficult to predict and not necessarily related to a common protocol framework. In the conducted tests: identical connection end-points, transmission proceeding at the same time, yet using different protocols; the paths differ significantly in SRTT (ipv4: 40–110 ms and ipv6: 60–300 ms) and number of hops (ipv4: 9 hops and ipv6: 17 hops). Moreover, as the ipv6 traffic traverses the potentially congested international backbone, it is more likely to be exposed to throughput fluctuations.

In this paper, extremum seeking control [14] is used as a basis to find a solution to problem (1) when the channels cannot be treated as independent of each other. The idea behind extremum seeking control is to obtain optimal system performance (according to a chosen measure) for an uncertain, possibly non-linear dynamic system by probing it through a known excitation injected into the control input. By observing the system response and the gradient of performance index it is possible to retrieve the unknown plant properties. In the typical implementation, the estimate of performance index gradient is obtained through a periodic excitation applied to linear system representation. Usually, this signal is much faster than the nominal plant dynamics. The response induced by the injected perturbation can then be removed by a low-pass filter (Fig. 2).

Owing to the hard non-linearities and delays, the classical extremum seeking approach cannot be directly applied to optimize the MPTCP transmission. Therefore, in this work, a modified method is elaborated. In the classical approach, by observing the performance index gradient, one estimates the unknown system parameters. In the method proposed here, the system parameters (path properties) are determined with high accuracy and thus assumed known, whereas the gradient measurement is used to decide whether the mutual dependency among the MPTCP sub-channels occurs, or not. The channels are deemed unrelated (independent) if the gradient is zero. Otherwise, an appropriate action needs to be taken, as illustrated in Fig. 2. In the discussed method, in order to assess the extent of mutual dependency of the sub-streams, the gradient of performance index (1) needs to be evaluated. It is assumed that the device establishes two paths only, as is the most common situation in the considered application area. The power efficiency is improved by changing the sub-channels load so

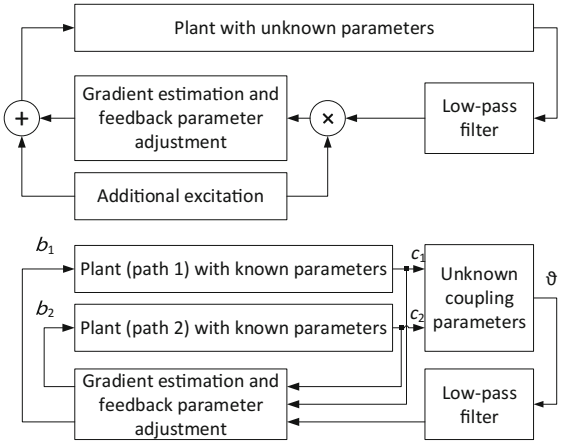


Fig. 2. Classical approach to extremum seeking control [14] (above) and the proposed one (below) with the additional excitation replaced by a legitimate signal from other channels.

that the gradient $\nabla\vartheta = \dot{\vartheta} \leq 0$. Since the performance index $\vartheta(t) = f(c_1(t), c_2(t))$, the chain rule applied to (1) leads to

$$\dot{\vartheta} = \frac{\partial\vartheta}{\partial c_1}\dot{c}_1 + \frac{\partial\vartheta}{\partial c_2}\dot{c}_2 \leq 0. \tag{4}$$

Assuming p_i and D change sufficiently slow, the energy expenditure related to channel i may be approximated as $\vartheta_i = \int_0^{T_i} p_i(t)dt \approx p_i T_i$. Then, rewriting (1) with $T_i \approx b_i/\mathrm{av}(c_i)$ and $\max T_i \approx (b_1 + b_2)/[\mathrm{av}(c_1) + \mathrm{av}(c_2)]$, one obtains

$$\vartheta = \rho_1 \frac{b_1}{\mathrm{av}(c_1)} + \rho_2 \frac{b_2}{\mathrm{av}(c_2)} + D_w \frac{b_1 + b_2}{\mathrm{av}(c_1) + \mathrm{av}(c_2)}, \tag{5}$$

where $\mathrm{av}(c_i) = f_i/\tau_i$ is the average throughput of channel i obtained as the fraction of *in-flight* (i.e., sent but not acknowledged yet) data (f_i) and SRTT (τ_i). D_w equals $D + w_1 + w_2$. The quantities f_i and τ_i are available in any contemporary SPTCP stack.

Applying multivariable chain rule (4) to (5), the gradient is determined as

$$\nabla\vartheta = \dot{\vartheta} = -\rho_1 \frac{b_1}{\mathrm{av}^2(c_1)}\dot{c}_1 - \rho_2 \frac{b_2}{\mathrm{av}^2(c_2)}\dot{c}_2 - D_w \frac{b_1 + b_2}{[\mathrm{av}(c_2) + \mathrm{av}(c_2)]^2}(\dot{c}_1 + \dot{c}_2). \tag{6}$$

The term $\rho_i b_i/\mathrm{av}^2(c_i)$ in (6) can be physically interpreted as the inverse of energy efficiency of path i and $D_w(b_1 + b_2)/[\mathrm{av}(c_1) + \mathrm{av}(c_2)]^2$, as the inverse of energy efficiency of the communicating device as a whole.

Since by definition ρ_i, c_i, b_i, and D_w are all positive and bounded, according to (6), the gradient follows the changes of the channel capacity (with reverse sign) and the energy dissipation decreases with $\left|\dot{\vartheta}\right|$ increasing. Therefore, the minimum

energy dissipation – the largest gradient – is achieved by transferring the data as fast as possible, i.e., when $\dot{c}_i \to \infty$. On the other hand, the channels have limited capacity and the derivatives in Eq. (6) cannot be positive all the time. In particular, when the common bottleneck is encountered, then the derivatives $\dot{c}_i$ differ in sign, and the gradient value is determined by the power efficiency of the transmitting device.

Remark 1. In some publications, e.g., [15], the use of correlation is advocated to assess the channel coupling. The correlation-based methods, however, require long measurement history to infer about the channel dependency due to temporal differences (mainly SRTT fluctuations). For the practical reasons, a method with short memory is preferable, as proposed in this work.

Reordering the terms in (6) yields

$$\begin{aligned} &-\rho_1 \tfrac{b_1}{\mathrm{av}^2(c_1)} \dot{c}_1 - \rho_2 \tfrac{b_2}{\mathrm{av}^2(c_2)} \\ &-D_w \tfrac{b_1}{[\mathrm{av}(c_1)+\mathrm{av}(c_2)]^2} (\dot{c}_1 + \dot{c}_2) - D_w \tfrac{b_2}{[\mathrm{av}(c_1)+\mathrm{av}(c_2)]^2} (\dot{c}_1 + \dot{c}_2) \leq 0. \end{aligned} \tag{7}$$

Thus,

$$\frac{b_1}{b_2} \geq -\frac{\rho_2 \frac{1}{\mathrm{av}^2(c_2)} \dot{c}_2 + D_w \frac{1}{[\mathrm{av}(c_1)+\mathrm{av}(c_2)]^2} (\dot{c}_1 + \dot{c}_2)}{\rho_1 \frac{1}{\mathrm{av}^2(c_1)} \dot{c}_1 + D_w \frac{1}{[\mathrm{av}(c_1)+\mathrm{av}(c_2)]^2} (\dot{c}_1 + \dot{c}_2)}, \tag{8}$$

if the denominator is positive and

$$\frac{b_1}{b_2} \leq -\frac{\rho_2 \frac{1}{\mathrm{av}^2(c_2)} \dot{c}_2 + D_w \frac{1}{[\mathrm{av}(c_1)+\mathrm{av}(c_2)]^2} (\dot{c}_1 + \dot{c}_2)}{\rho_1 \frac{1}{\mathrm{av}^2(c_1)} \dot{c}_1 + D_w \frac{1}{[\mathrm{av}(c_1)+\mathrm{av}(c_2)]^2} (\dot{c}_1 + \dot{c}_2)}, \tag{9}$$

otherwise.

In the steady state, when $\dot{c}_i \to 0$, applying L'Hôpital's rule to (8) (or (9)), one obtains

$$\frac{b_1}{b_2} = -\frac{\rho_2 \frac{1}{\mathrm{av}^2(c_2)} + D_w \frac{1}{[\mathrm{av}(c_1)+\mathrm{av}(c_2)]^2}}{\rho_1 \frac{1}{\mathrm{av}^2(c_1)} + D_w \frac{1}{[\mathrm{av}(c_1)+\mathrm{av}(c_2)]^2}}. \tag{10}$$

Inequalities (8)–(10) define the scheduler for coupled channels. If $b_1 > b_2$ the current segment ought to be transmitted by channel 1, and by channel 2, otherwise.

5 Implementation Issues

For the purpose of verification of the developed load-balancing strategies a new Linux kernel module implementing (8)–(10) has been created. In addition to this new scheduler, the Linux implementation encompasses the standard path-manager that establishes the paths along which the data will be directed and MPTCP master controller that targets the fairness issues. Since the objective of

this work is to provide a method of stream division, only the scheduler has been modified with respect to the reference MPTCP implementation [6]. The other modules are left unaltered with the parameters set at their defaults, i.e., the path manager operates in the full-mesh mode (the number of sub-streams is the product of the number of logical interfaces at both peers) and no specific master control is executed. The standard Linux congestion control algorithm (CUBIC) supervises the individual SPTCP sub-streams.

In contrast to the default scheduler, in the proposed approach, the previously selected paths influence the present scheduling decisions. To properly evaluate (8)–(10), the scheduler uses the internal SPTCP variables (f_i and τ_i). In addition, it needs to track variable c_i to obtain $\dot{c}_i$. For that purpose, a recurrent filter (typical for the TCP implementations) is used, namely,

$$\dot{c}_i(k) = (1-\alpha)\,\dot{c}_i(k) + \alpha\,(c_i(k) - c_i(k-1))\,, \tag{11}$$

where k denotes the subsequent time instants of $\dot{c}_i$ evaluation. The averaging coefficient α has been experimentally chosen as 1/16. In order to obtain the allocation decision, scheduler (8)–(10) performs two iterations per MPTCP segment (*mptcp_for_each_sk* loop): the first one provides the sub-stream coefficients ($\dot{c}_i$), the second one finds the best channel for data transfer. The details of the kernel module organization and power measurement one can find in [13], where the case of independent channels has been considered. In the tests described here in Sect. 6, the following artificially injected patterns have been applied: (a) constant, (b) linear increase/decrease (simulates moving away, or getting closer to a link layer hub), (c) stepwise (simulates handover, or getting in or out of a radio shadow), and (d) oscillatory (simulates channel fading). Consequently, in addition to already covering a broad spectrum of practical networking situations, new profiles can be seamlessly introduced into the applied framework for conducting a variety of tests.

6 Evaluation

The developed scheduling solution has been verified using the setup based on the general system architecture depicted in Fig. 1. The popular low-end Raspberry Pi device has been chosen as the MPTCP client and a high-end virtual machine in the local data center (in the *cloud*) as the server, which corresponds to the typical use pattern in the MPTCP networking. During all the tests the server runs unmodified, following the standardized MPTCP rules. In all the experiments, the public network (neither Intranet, nor closed simulations) is employed. While in such environment a single measurement is hardly repeatable, a large number of trials allows one to filter the artifacts and gives a sound basis for the protocol assessment in a realistic rather than artificially created lab environment. To minimize the influence of channel temporal variations and give statistically meaningful results, the tests have been repeated multiple times.

6.1 Test Scenarios

Two channel combinations have been tested. In the first one – scenario 1 – the probability of a common bottleneck is low, in the second – scenario 2 – high. The first scenario represents the case of a smartphone, or an IoT device [16,17], communicating with the server, with one interface connected to a cable network and the second inter-face connected to an LTE network. The cable and LTE networks are unrelated. The measured initial SRTT of the cable network channel varies within 70–85 ms and the segments traverse 7 hops, whereas for the LTE connection the SRTT fluctuates within 80–400 ms and the segments encounter 9 hops before reaching the destination. In the second scenario, the same interface is used by the endpoints, yet the channels differ in the applied network protocol. Such paths frequently influence each other. During the tests, the first path consists of 9 hops and the measured SRTT varies in the interval 40–110 ms and the second path covers 17 hops with the SRTT fluctuating in the range 60–300 ms.

6.2 Test Case Conditions

In all the tests, $D_w = 600$ [mW], $\rho_1 = 30$ [mW/Mb], $\rho_2 = 400$ [mW/Mb] (scenario 1) and $\rho_1 = \rho_2 = 140$ [mW/Mb] (scenario 2). A single test case (in both scenarios) involves an *iperf3* generated stream controlled first by the default scheduler, next by the proposed energy-aware one. The tests are performed at different days and times, and using different power profiles. As already mentioned, in order to obtain statistically relevant results the tests are repeated multiple times and are executed at different times and days. During all the tests the gain is defined as

$$\mathrm{G} = \left(\vartheta^{\mathrm{default}} - \vartheta^{\mathrm{proposed}}\right) / \vartheta^{\mathrm{default}}, \tag{12}$$

where $\vartheta^{\mathrm{default}}$ is the energy dissipated when the default scheduler operates, whereas $\vartheta^{\mathrm{proposed}}$ refers to the energy expenditure of scheduler (8)–(10).

6.3 Test Results

Selected test results are presented in Table 1. The advantages of using the proposed power-aware scheduler are highlighted when the amount of fluctuations (congestion incidents, interferences, etc.) in the network grows. The advantages of the discussed solution are particularly visible during rush hours (7–8 PM). In scenario 1, the proposed scheduler outperforms the default one in terms of lower energy expenditure (10–36%), and occasionally in terms of throughput. In an uncommon situation, when the channel with a shorter SRTT dissipates more energy (LTE), it is possible to obtain the gain as high as 90%. In scenario 2, the power-aware scheduler gives 4–10% better results (maximum 12%) with respect to the reference one.

Table 1. Experiment results. T – Time of transmission [s], ϑ – dissipated energy [J], G – gain in applying energy-aware solution [%]. R – rush hours (7–8 PM), E – Early morning (5–6 AM), O – ordinary hours (1 PM–4 PM).

Scenario 1					Scenario 2					When
Default		Proposed			Default		Proposed			
T	ϑ	T	ϑ	G	T	ϑ	T	ϑ	G	
8.58	7.99	7.99	7.99	**8**	9.56	6.45	9.50	6.41	**1**	E
8.20	8.01	8.01	8.01	**11**	9.52	6.37	9.45	6.35	**0**	E
9.35	9.02	9.02	9.02	**3**	10.62	7.16	10.07	6.91	**4**	O
9.82	8.34	8.34	8.34	**14**	10.66	7.19	10.20	6.97	**3**	O
11.59	9.12	9.12	9.12	**21**	9.95	6.60	9.72	6.56	**1**	O
22.55	16.1	16.1	16.1	**23**	13.07	7.71	11.41	7.06	**8**	R
22.82	15.5	15.5	15.5	**36**	12.93	7.55	11.52	7.12	**6**	R

During ordinary hours (12 AM–4 PM), in scenario 1, the gain of using the energy-aware schedulers oscillates between 3% and 25% and differs much in subsequent test runs. Nevertheless, a case when the gain is negative has not been observed. In scenario 2, the power saving is smaller gain is smaller than in rush hours yet still meaningful with respect to the default solution.

When the network traffic is stable and low (e.g. in the early morning – 5–6 AM) the proposed scheduler provides limited gain. The allocation decisions are highly influenced by the buffer-bloat phenomenon (BB). While other network users are absent, large buffers in the intermediate devices prohibit the *cwnd* reduction at the end-points and SRTT increases. When the default scheduler controls the segment-to-path assignment, the energy greedy interface (LTE) is erroneously chosen over the more economic one if the SRTT, elongated by BB, exceeds the transfer time in the expensive one. The proposed scheduler is immune to the BB effects – see also [18]. When the throughput does not undergo significant variations ($\dot{c}_i$'s are close to zero), then the proposed scheduler operates in the steady-state conditions according to (10). Except for the segment ordering there are no substantial differences comparing to the reference solution.

6.4 Detailed Analysis

An example run of scheduler (8)–(10) under scenario 2 is illustrated in Fig. 3. The established paths have similar initial SRTT and they are exposed to BB (SRTT grows from the initial 52 ms for channel 1 and 69 ms for channel 2, respectively, up to 400 ms as a result of buffering in the intermediate routers). At the beginning of the transmission, the scheduler is in the steady state (the split ratio is constant) and it allocates the channels according to (10). At $t \approx 4$ s, stream 2 experiences a congestion incident. Both the *in-flight* data (f_2) and throughput (c_2) are reduced, thus $\dot{c}_2 < 0$. Then, according to rule (9),

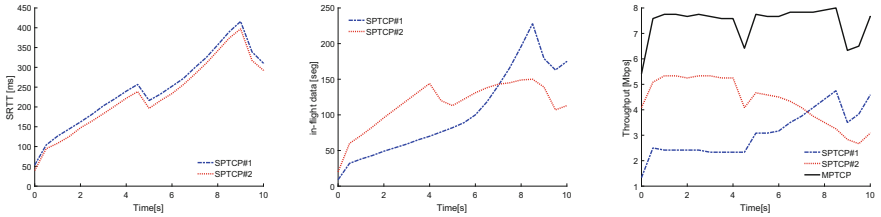


Fig. 3. Performance of the proposed scheduler in scenario 2, sampling period 0.5 s: SRTT (left), in-flight data (middle), throughput (right).

the transfer speed in stream 2 is throttled, and in stream 1 it accelerates. Since the sum $\mathrm{av}(c_1) + \mathrm{av}(c_2)$ is approximately constant, the stream 2 cannot increase its speed and the transmission proceeds mainly in channel 1.

6.5 Beneficial Side Effects

As a beneficial side effect, it has been noticed that even if the power coefficients are not considered in the allocation decisions ($\rho_1 = \rho_2 = 1$, $D = D_w = 0$ in (8)–(10)), the energy-aware scheduler gives on average a few-percent improvement over the default one. This gain increases with the throughput fluctuations. It is attributed to the BB reduction (as noticed also in the case of SPTCP in [18]). Moreover, without any tricky heuristics, or link layer dependency [6], the jitter is constrained to 2–4 segments and fast response to the channel variability is obtained. In consequence, the redundant scheduler from the standard implementation [6] need not be applied, low buffer occupancy is observed, and the risk of head-of-line blocking is limited. All these favorable characteristics are obtained at a small computational cost.

7 Summary and Conclusions

The paper presents a new stream splitting algorithm for the MPTCP traffic scheduler for the paths encountering a common bottleneck. In the proposed solution, the load over the available network interfaces is distributed so as to achieve low energy consumption. Thus, it is particularly well-suited for mobile, battery-powered devices, like tablets, or smartphones. The algorithm is formulated on the basis of a modified extremum seeking approach. By incorporating broader information about the channel dynamic characteristics, the energy efficiency as compared with the default, recommended solution, is improved. Numerous experiments (conducted in the open, public Internet, as opposed to the commonly-applied simulated environments) show several percent reduction in the energy expenditure. In many of the analyzed cases, higher throughput is also attained. As a result, the energy economy of the multipath transfer in a dynamic networking environment can be enhanced.

References

1. Ignaciuk, P., Bartoszewicz, A.: Congestion Control in Data Transmission Networks: Sliding Mode and Other Designs. Springer, London (2013). https://doi.org/10.1007/978-1-4471-4147-1
2. Ignaciuk, P., Karbowańczyk, M.: Active queue management with discrete sliding modes in TCP networks. Bull. Polish Acad. Sci. Tech. Sci. **62**(4), 701–711 (2014)
3. Peng, Q., Walid, A., Hwang, J., Low, S.: Multipath TCP: analysis, design, and implementation. IEEE/ACM Trans. Netw. **24**(1) 596–609 (2016)
4. Xu, C., Zhao, J., Muntean, G.: Congestion control design for multipath transport protocols: a survey. IEEE Commun. Surv. Tutor. **18**(4), 2948–2969 (2016)
5. Ford, A., Raiciu, C., Handley, M., Bonaventure, O.: TCP extensions for multipath operation with multiple addresses. Technical report 6824, RFC (2013)
6. Barré, S., Paasch, C.: Multipath TCP Linux Kernel implementation. http://www.multipath-tcp.org
7. Kwak, J., Choi, O., Chong, S., Mohapatra, P.: Processor-network speed scaling for energy-delay tradeoff in smartphone applications. IEEE Trans. Netw. **24**(3), 1647–1660 (2016)
8. 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)
9. 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)
10. 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)
11. 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)
12. 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)
13. Morawski, M., Ignaciuk, P.: On implementation of energy-aware MPTCP scheduler. In: Borzemski, L., Świątek, J., Wilimowska, Z. (eds.) ISAT 2017. AISC, vol. 655, pp. 242–251. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-67220-5_22
14. Ariyur, K., Krsti, M.: Real-Time Optimization by Extremum-Seeking Control. Wiley, Hoboken (2003)
15. Kou, L., Liu, J., Hu, M.: A multiple-path TCP congestion control algorithm based on sub flow correlation matrix. In: Proceedings on International Conference on Cloud Computing and Internet of Things, Changchun, China, pp. 140–144 (2014)
16. Morawski, M., Ignaciuk, P.: Reducing impact of network induced perturbations in remote control systems. Control Eng. Pract. **44**(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)
18. Cardwell, N., Cheng, Y., Gunn, C., Yeganeh, S., Jacobson, V.: BBR: congestion-based congestion control. Commun. ACM **60**(2), 58 (2017)

Light-Weight Congestion Control for the DCCP Protocol for Real-Time Multimedia Communication

Robert R. Chodorek[1(✉)] and Agnieszka Chodorek[2]

[1] Department of Telecommunications, The AGH University of Science and Technology, Al. Mickiewicza 30, 30-059 Krakow, Poland
chodorek@agh.edu.pl

[2] Department of Information Technology, Kielce University of Technology, Al. Tysiaclecia Panstwa Polskiego 7, 25-314 Kielce, Poland
a.chodorek@tu.kielce.pl

Abstract. The Datagram Congestion Control Protocol (DCCP) is a multi-purpose transport protocol that does not preserve reliability of transmission. The DCCP was intended to convey large amounts of data, especially in the real-time - which makes it suitable for real-time multimedia applications. Consequently, the RTP/DCCP protocol stack can be used for data transmission (instead of the RTP/UDP protocol stack, typically used during the transmission of real-time multimedia), when the DCCP works in mode CCID 3 (TCP-Friendly Rate Control, TFRC).

However, the TFRC manifests strong equality towards competing TCP flows, which (in some situations) can lead to the degradation of the multimedia stream transmitted over the TFRC. In this paper we describe the light-weight congestion control mechanism, previously designed by the Authors for TFRC-based real-time multimedia communication. The mechanism is based on the TFRCs congestion control, in which the original TFRCs throughput equation was substituted with a linear throughput equation. The mechanism that was introduced for the DCCP implementation and the results of experiments carried out in a mixed (real and emulated) network show that this type of congestion control is more suitable for multimedia than the DCCP working in the standard CCID 3 mode.

Keywords: Multimedia · Real-time · DCCP · Congestion control

1 Introduction

Modern communication is based on convergent, multi-service networks, where data traffic and multimedia traffic coexist in shared links. These two kinds of traffic have two different quality of service (QoS) requirements, as well as two different approaches to the problem of network congestion. As a result, when a window-controlled data stream meets with a multimedia stream, which cannot

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 52–63, 2018.
https://doi.org/10.1007/978-3-319-92459-5_5

be window controlled, the stream may not be congestion controlled at all. Such multimedia streams are called unresponsive, in that they cannot respond to congestion. If an unresponsive flow meets a congestion controlled (responsive) flow in the same link, the unresponsive flow shows a tendency to dominate the transmission in the link. It was observed that in the case of Transmission Control Protocol (TCP) transmissions, multimedia can overwhelm the traffic for TCP flows. Lack of TCP-like congestion control was blamed for this effect, so protocols that implements TCP-like congestion control are called TCP-friendly protocols.

For the last few years, because of a significant amount of multimedia traffic in overall communication traffic, newly designed protocols are often required to be TCP-friendly [2,9,16,17]. Nowadays, web browsers that use the Real-Time Communications (WebRTC) technologies enable congestion control for transmission of multimedia data, based on the TCP-Friendly Rate Control (TFRC) [7] congestion control mechanism [10].

However, obligatory TCP-friendliness can cause problems in the case of real-time multimedia transmissions carried out in medium and heavily congested networks. The aim of this paper is to analyze a light-weight congestion mechanism intended for the Datagram Congestion Control Protocol (DCCP) working in the TFRCs congestion control mode. The mechanism was designed for real-time multimedia communication in medium and heavily congested networks. It is an implementation of our previous work [1], in which we substitute a linear throughput equation for the original TFRCs throughput equation in the DCCP protocol.

The paper is organized as follows. The next Section is devoted to the DCCP protocol. In Sect. 3 we describe the light-weight congestion control mechanism implemented for the DCCP protocol that conveyed real-time multimedia. Section 4 shows the experimental environment that combines real network equipment and an emulation server. Section 5 presents results of the experiments carried out in the mixed, real and emulated environment. Section 6 summarizes our experiences.

2 The DCCP Protocol and Its Use for Multimedia Transmission

Typically, real-time multimedia transmission is carried out using the User Datagram Protocol (UDP), which in the case of professional and semi-professional applications is often supplemented by the Real-time Transport Protocol (RTP), working on top of the UDP, and using its checksum and multiplexing services. The transport layer is divided into two sublayers, where the RTP occupies the upper sublayer, and the UDP the lower one. Because of the lack of congestion control in both transport protocols[1], the Datagram Congestion Control Protocol, specified in the RFC 4340 [13], is intended to be used instead of the UDP.

[1] The UDP is not congestion controlled at all. The RTP also isn't congestion controlled, however it supports media translators, which usually are placed at the boundary of high-speed and slow network. Translators lower bit rate of transmitted stream and make it available for low-speed clients.

The DCCP is a multi-purpose transport protocol, designed to convey a large amount of data, such as the UDP. In contrast to the UDP, which is able to create both unicast and multicast connections, the DCCP transmits data only via unicast connections. Both the UDP and the DCCP do not assure reliable data transfer[2], which makes the protocols suitable for streaming media and other real-time applications.

The DCCP doesn't support specific multimedia services (such as, for example, timestamps or source identifiers), so professional and semi-professional use of the DCCP for multimedia transmission requires the co-operation of the protocol with other transport protocols, that are able to assure the necessary functionalities. If the DCCP works with RTP-transported multimedia, the protocol will replace the UDP in the RTP/UDP protocol stack and the RTP/DCCP protocol stack is constructed.

Its worth noting that the replacement of the UDP by the DCCP limits the functionality of the overlaying protocol. The RTP is a multicast protocol, but the underlying DCCP - due to its unicast nature - confines it to point-to-point transmissions. This is not noticeable in the case of naturally unicast services, such as video telephony or Video on Demand (VoD), but multicast multimedia services must substitute one multicast connection with a series of unicast ones.

The DCCP implements multi-mode congestion control of unicast connections. Each congestion control mode is identified by a Congestion Control Identifier (CCID). The specification of the protocol assigns only CCID 2 and 3, while CCID 0–1 and 4–255 are reserved for future applications. Both modes offer TCP-friendly congestion control, which is able to use both implicit (through packet loses) and explicit (through ECN signalling) congestion notification. However, they achieve TCP-friendliness of DCCP flows with the use of different methods of building TCP-like congestion control:

- direct implementation of TCP's congestion control mechanism,
- emulation of TCP's behavior with the use of mathematical modelling.

The first method directly implements the mechanism specified in the RFC 2581 [18] document - including the congestion control window, two areas of probability (low and high probability of congestion increases) associated with the mechanisms of slow start and exponential window growth, etc. This method allows applications to maximally utilize available bandwidth and is identified as mode CCID 2, specified in the RFC 4341 [5].

The second method, typical for TCP-friendly multimedia systems, describes TCP's behavior under congestion using the so-called TCP throughput equation, and sends packets of TCP-friendly protocol according to the rate computed from this equation. The best known TCP-friendly congestion control module, based

[2] The UDP is not error controlled at all. The DCCP assures only reliable negotiation of options and reliable congestion notification, as well as transmits full signalling information about packet losses.

on the throughput equation, is the TFRC [7]. According to the equation used by the TFRC, the throughput of the TCP connection is a function of a packet error rate (PER) [1,7]:

$$T(PER) = \frac{MSS}{RTT} \cdot C \cdot \cdot \left(\sqrt{\frac{2}{3} \cdot PER} + 12 \cdot PER \cdot \sqrt{\frac{3}{8} \cdot PER} \cdot \left(1 + 32 \cdot PER^2\right) \right)^{-1} \quad (1)$$

where T is a TCP throughput, MSS is TCP's maximum segment size, RTT is round-trip time, and C is a scale coefficient.

The DCCP implements Eq. (1) as mode CCID 3, described in detail in the RFC 4342 [6].

The last mode, CCID 4, is a modification of the TFRC mechanism for transmission of small packets (e.g. voice stream). A profile for Congestion Control Identifier 4 is specified in the RFC 5622 [8]. Modes CCID 3 and CCID 4 are recommended for the transmission of streaming media.

3 Light-Weight Congestion Control for the DCCP Protocol: Design and Implementation

Although the TFRC is recommended as the congestion control building block for protocols that convey real-time multimedia, the TFRC congestion control mechanism manifests strong equality towards competing TCP flows. This feature is an effect of the TCP-friendliness of the TFRC - therefore, it is desirable. However, in some situations, it can lead to the degradation of multimedia streams transmitted over the TFRC.

This soft nature of the TFRC was moved to the DCCP working in mode CCID 3. Moreover, the DCCP that was used as a lower sublayer of the transport layer in the RTP/DCCP stack, limits the aggressiveness of the RTP. As a result, the TCP becomes too aggressive to allow the RTP/DCCP to preserve QoS for multimedia traffic. In this paper we propose a new mode of the DCCP congestion control, based on the modified TFRC building block described in [1], which joins elements of RTP behavior with equation-based congestion control and rate control mechanism of the TFRC.

RTP streams are unresponsive. Their response to congestion is not very elastic, as in the case of congestion controlled protocols. The RTP flow itself is not able to react to congestion, so the only reaction is packet losses in intermediate routers. Packet losses reduce the bit rate of the flow linearly [1]. As a result, in a network that is well-dimensioned for multimedia traffic, the RTP throughput equation depends only on the bit rate of the carried multimedia stream and the packet error rate [1]:

$$T(PER) = BR(1 - PER) \quad (2)$$

where BR is a bit rate of multimedia stream.

The light-weight congestion control mechanism for the DCCP protocol has been implemented in an environment of the Linux operating system. The Linux kernel since version 2.6.x contains the implementation of the DCCP protocol (stored in branch /net/dccp/) along with congestion control mechanisms implemented for mode CCID 2 and CCID 3 (stored in branch /net/dccp/cids/). The protocol's implementation is build as a kernel module (dccp). Depending on the initial configuration of the Linux kernel, the implementation can be loaded one of two ways: immediately (loaded by the system during the start of the Linux) or on demand (loaded by the user). The congestion control CCID 2 and 3 modes are implemented as a separate modules (dccp_ccid2 and dccp_ccid3, respectively).

The original DCCP protocol implementation on Linux was written by Arnaldo C. Melo. It uses standard socket application programming interface (API). Therefore, the code of user programs, written for server and client, is similar to the code of a regular TCP application. Some parts of the DCCP implementation share code with the Linux's TCP implementation and any other Linux's implementation of INET[3] transport protocols, such as the Stream Control Transmission Protocol (SCTP). In the case of the DCCP implementation, the server initializes a socket, binds it to a port, and waits to accept clients to connect. The client needs only to initialize a socket and connect to the server. This model of connection setup is similar to the TCP's connection setup.

After connection setup, a client and a server are able to exchange data. If DCCP's congestion control mechanism recognizes that the sending rate is too large, the current data packet will be refused before it can be sent to a network. Thus, the user's programs that use the DCCP must check the status of each sending routine.

Our implementation for the new DCCP's congestion control module has been created for the Linux 4.4.0 kernel. We built a new Linux kernel module, analogous to existing modules for CCID 2 and 3 dccp_ccid_lin. Our implementation included a base congestion control module (ccid.c), which registers and calls the currently used congestion control module. Parameters of both the DCCP and congestion control modules are set by variables exported from the dccp_ccid_lin module and the basic dccp module. This method enables (re)configuration of kernel module parameters at runtime.

The basic data needed for proper working of the dccp_lin module are stored in the ccidlin_hc_tx_sock structure. The structure contains, for example, current sending rate (tx_x), receive rate (tx_x_recv), current loss event rate (tx_p), mean packet size, expressed in bytes (tx_s), estimate of current round trip time (tx_rtt), target bit rate of transmitted multimedia stream (tx_tbr).

At the very beginning the current sending rate tx_x must be set to the tx_tbr value. The tx_tbr can be set as one of the kernel module parameters or by the additional socket option. During transmission, when the DCCP packet is processed, the function ccidlin_hc_tx_update_x is called. This function updates the current sending rate tx_x. If the transmission is inactive for idle period (more than 2*RTT or more than an Application-Limited Periods [6]) then the current

[3] A group of internet protocols in the Linux system.

sending rate tx_x is set to the initial sending rate. If errors occur, the current sending rate tx_x is modified according to the formula (2).

4 Test Environment

Experiments were carried out using a mixed environment, depicted in Fig. 1, where real fragments of a test network are combined with the emulated one. The emulated fragment of the test network is marked in gray in Fig. 1. As the network emulator, the Berkeleys ns-2 simulator [4] working in emulation mode was used. The choice of the emulator is guided by its light-weight packet processing methodology (only chosen header information is processed in emulated nodes), which is characteristic of simulators. Fast, light-weight packet processing allows the ns-2 for emulation of both more network nodes and more high-speed connections than typical emulators, such as the ns-3, using the same emulation server. For the emulation server, a dual processor machine, equipped with two Intel Xeon processors and six Gigabit Ethernet interfaces, was used. The original ns-2 software has been supplemented by extension [3,14], which allows the ns-2 to better emulate real-time multimedia traffic, as well as by extension [15], designed to use an external switch (switches S1 and S2 in Fig. 1.) as a traffic expander. Standard DCCP protocol, implemented in the Linux kernel, was extended with improvements described in the Sect. 3, which enable usage of the linear throughput equation.

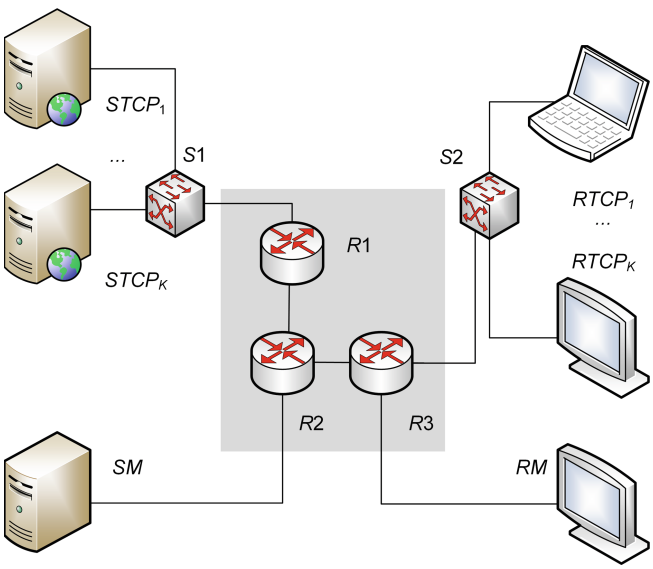


Fig. 1. Topology of test network

Experiments were performed using a single-bottleneck topology. Real fragments of the test network are connected to the emulation server using Gigabit Ethernet technology. Almost every link of emulated fragment of the test network had throughput of 1 Gbps and propagation delay of 1 μs. The only exception is the bottleneck link, which connects routers R2 and R3. Parameters of this link are is assumed to be 100 Mbps of throughput and 3 ms of propagation delay. This topology makes the router R2 the congested node in most of experiments carried out.

Variable Bit Rate (VBR) video stream was transmitted between SM and RM end-systems and the target bit rate of the stream was equal to 40 Mbps (the maximum for the Blu-ray), i.e. the network is well-dimensioned for multimedia. In all experiments, a collection of eight High Definition Television (HDTV) video sequences, encoded to the H.264 standard and publicly available at [19], were used. Each clip belonging to the sequence lasts for 19 s and includes 1920 x 1080 native video, captured at 30 frames per second [12]. These video sequences are owned by NTIA/ITS, an agency of the U.S. Federal Government, and were created under Project Number 3141012-300, Video Quality Research, in 2008.

The media server (SM) was build using the VLC [20] software tool, working under the control of the Linux operating system. Some bug fixes and improvements of standard vlc to enable proper co-operation with the DCCP were made. Real-time video transmissions, which generates foreground traffic, were carried out using following variants of the DCCP protocol:

- the DCCP with TFRC congestion control (CC) mechanism, in other words the DCCP working in the standard CCID 3 mode (curves labeled DCCP TFRC CC in Figs. 2 and 3),
- the DCCP with light-weight congestion control mechanism, which use linear throughput equation (curves labeled DCCP light-weight CC in Figs. 2 and 3),
- the DCCP with TCPs congestion control mechanism, i.e. the DCCP working in CCID 2 mode (curves labeled DCCP TCP's CC in Figs. 2 and 3).

For the sake of comparison, transmissions with the use of typical RTP/UDP protocol stack also were carried out (curves labeled RTP in Figs. 2 and 3).

As a background traffic, a number of TCP transmissions are carried out with the use of the iPerf tool [11], also working under the control of the Linux system. Senders of background traffic (from $STCP_1$ to $STCP_K$ in Fig. 1) are connected to the router R1 using the traffic expander S1, and receivers of TCP traffic ($RTCP_1$ to $RTCP_K$) are connected to router R3 through the traffic expander S2. Transmissions are performed between the pair of nodes $STCP_i$ and $RTCP_i, i = 0, ..., K$. If the total number of TCP flows K is close to 0, the network will not be congested. The growth of number of flows causes growth of congestion.

5 Results

During all of the experiments throughputs achieved for video and bulk data transmissions were measured and the number of packet drops were recorded.

The number of background TCP flows were changed from 0 to 10. The number of foreground video streams was equal to one, but (to avoid the influence of individual video clips on overall results) the transmitted stream was made of clips randomly chosen from sequences [19], and the transmission of the currently chosen clip started from a randomly chosen video frame. Each experiment was repeated over 10 times and the results were averaged.

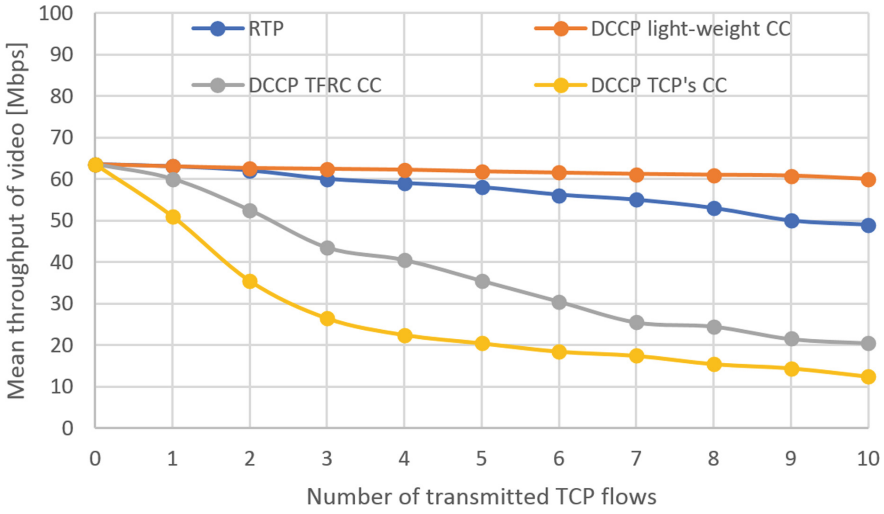


Fig. 2. Mean throughput of video stream

Figures 2 and 3 presents throughput of both the foreground video stream and the background TCP flows as a function of the number of TCP flows K. K equal to zero denotes that the bottleneck link was dedicated to the video transmission (the video stream did not have to share this link with any background TCP flow). Because real-time video transmission is rate-limited (transport protocol cannot send more data than the amount of data generated in real-time), the throughput of the video transmission achieved for $K = 0$, i.e. 63.5 Mbps, is the maximum.

As is depicted in Fig. 2, the presence of background trafficreduces the throughput of the video stream transmitted by the RTP protocol working over the UDP. In conditions as described in the previous Section, one competing TCP flows reduces mean video throughput by 0.5 Mbps (less than 1%), two TCP flows by 1.5 Mbps (about 2.5%), three by 3.5 Mbps (more than 5%), and ten competing TCP flows reduces mean video throughput by 14.5 Mbps (23%).

The mean throughput of the video stream transmitted with the use of the DCCP protocol heavily depends on the applied method of congestion control. The DCCP that works in CCID 2 mode (exact implementation of the TCP's CC) is typical for the TCP fair bandwidth allocation. The throughput of the

bottleneck link is divided amongst all foreground and background flows and streams that share this link. As a result, the DCCP is not able to preserve the real-time character of the video stream even in the case of a lightly congested bottleneck link. Even one competing TCP flow was already enough to reduce the mean throughput of the video stream by more than 10 Mbps (to 51 Mbps, which gives 80% of the mean throughput of the video stream in the dedicated link, $K = 0$). Two background flows competing for bandwidth with one video stream reduces the video throughput by another 15 Mbps, and the video stream achieves throughput of 35.5 Mbps (56% of 63.5 Mbps obtained for $K = 0$). In the case of $K = 10$, the throughput of the video stream collapses to 12.5 Mbps (20% of value measured for $K = 0$). It is worth mentioning here that CCID 2 is not recommended for multimedia, and the results are only comparative.

The DCCP congestion control mode CCID 3, which implements TFRC congestion control, is intended for multimedia transmission. The replacement of the exact implementation of TCP's congestion control by the approximation of its behaviour, given by the formula (1), which causes "reasonable fairness" of the TFRC competing for bandwidth with TCP flows, improves behaviour of the DCCP protocol during transmission of video streams. One TCP flow that shares the bottleneck link with the DCCP working in mode 3 is able to reduce the mean throughput of the video by only 3.5 Mbps (instead of 12.5 Mbps observed during the transmission with the use of CCID 2). However, it is still much worse than in the case of the RTP transmission, where a reduction of 3.5 Mbps was only seen when K = 3. Initially the rate of the descent of the throughput curve is smaller than in the case of the CCID 2, but when congestion grows the rate of descent also grows and curves plotted for CCID 2 and CCID 3 are close to each other. Generally, the condition of reasonability is satisfied. Throughput of the video stream observed for CCID 3 is within the range of $1 * T_2(K)$ to $2 * T_2(K)$, with a median of $1.58 * T_2(K = 8)$ and two mode values ($1.48 * T_2(K)$, $K = 2$ and $K = 9$; and $1.64 * T_2(K)$, $K = 3$ and $K = 10$, where $T_2(K)$ is a throughput of the video stream observed for CCID 2 when the number of competing TCP flows is K. The biggest difference between throughput of CCID 3 and CCID 2 were observed for $K = 4$ (the ratio between throughputs of CCID 3 and CCID 2 achieves 1.8), $K = 5$ (a ratio of 1.73) and $K = 6$ (a ratio of 1.65).

Video transmissions are rate-limited, and (to achieve real-time conditions of transmission) videos transfer rates must not exceed their rate limits. Fairness, as well as reasonable fairness toward competing flows, means that the work in the real-time regime is only possible to a limited extent. As a result, the DCCP working in congestion control mode CCID 3 is able to replace the UDP in the RTP/UDP protocol stack only if a network is lightly congested. Replacement of the TCP throughput Eq. (1) by the linear RTP's Eq. (2) allows congestion control mechanisms to have a more aggressive behaviour and for the resignation of equality, especially problematic for multimedia.

An effect of the use of the light-weight congestion control based on the linear throughput Eq. (2) is a flat, quasi-linear curve of video throughput shown in Fig. 2 (the curve labeled DCCP light-weight CC). In the case of non-congested

(K = 0) and lightly congested ($K = 1$) networks, throughput of RTP over DCCP video transmission is exactly the same as throughput obtained when the RTP over UDP protocol stack was used. From $K = 2$, the rate of descent of the RTP/DCCP throughput curve (from 0.1 to 0.4 Mbps per additional TCP flow; median: 0.3, mode: 0.2) is significantly smaller than the rate of descent of the RTP/UDP throughput (typically 1–2 Mbps per additional TCP flow). Only in the last case, K = 10, the heavily congested network caused a reduction of video throughput comparable with that observed for the RTP/UDP (0.8 Mbps vs. 1 Mbps). However, in the case of K = 10, the value of the throughput of the video stream transmitted by the DCCP that used the light-weight CC is 60 Mbps, which means that 10 TCP flows have reduced the video throughput only by 3.5 Mbps (5.5%). The same reduction of throughput (by 3.5 Mbps) was observed for the RTP over DCCP video transmission when $K = 3$ and for the DCCP working in congestion control mode CCID 3 when $K = 1$.

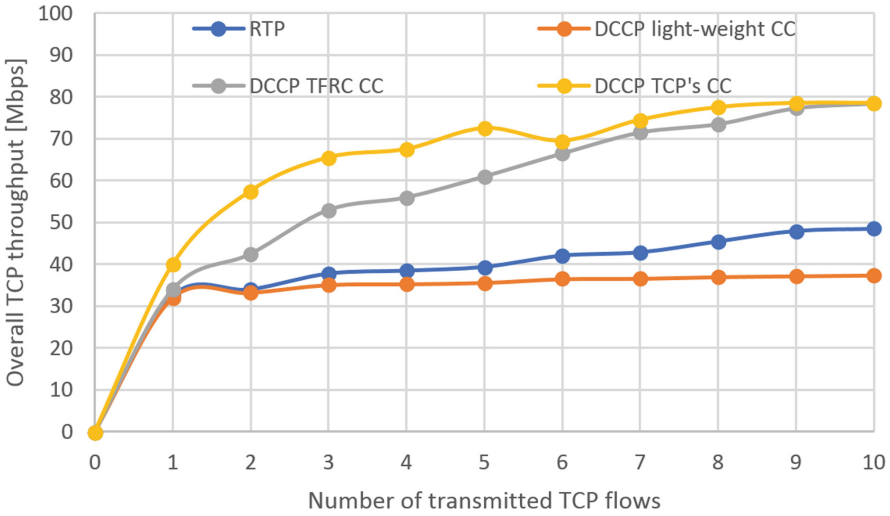


Fig. 3. Overall throughput of K TCP's flows

The mean throughput of the transmitted video (Fig. 2) determines the bandwidth available for background traffic. Thus, it has a major impact on the overall throughput of TCP flows (Fig. 3). As was to be expected, the largest overall throughput of TCP flows were observed when the classic TCP's congestion control mechanism was implemented, with its key feature of fair bandwidth allocation. The overall throughput of the TCP ranges from 40 Mbps ($K = 1$) to 78.5 Mbps ($K = 10$). The TCP-friendly DCCP congestion control mode CCID 3 (TFRC) also achieves good results, although (because of its worse, from the TCP point of view, "reasonable" fairness) not as good as pure TCP congestion control (CCID 2). If congestion increases, the gap between them closes slowly

from 15 Mbps ($K = 2$), through 12.5 Mbps ($K = 3$) to 11.5 Mbps ($K = 4$ and $K = 5$). If K is larger than or equal to 6, the curves of throughput plotted for the two DCCP's TCP-friendly congestion controls are close to each other. The gap between TCP's CC curve and the TFRC CC curve is reduced to 3 Mbps ($K = 6$ and $K = 7$), 1.2 Mbps ($K = 9$), and, finally, to 0.1 Mbps ($K = 10$).

Initially mean throughput of a video stream transmitted with the use of the RTP/UDP protocol stack and the RTP/DCCP protocol stack, where the DCCP uses the light-weight CC, is comparable (Fig. 2). As a result, the gap between throughputs of TCP background traffic, observed for two RTP-based transmissions (Fig. 3), is very small (less than 1 Mbps: 0.5 for $K = 1$ and 0.8 for $K = 2$). If congestion increases, this gap opens slowly from 2.7 Mbps ($K = 2$), through 3.8 and 5.6 Mbps ($K = 5$ and $K = 6$), and finally exceeds 10 Mbps (10.8 Mbps for $K = 10$ and 11.1 Mbps for $K = 10$).

6 Conclusions

The DCCP protocol works in different congestion control modes, two of which (CCID 3 and CCID 4) are intended for multimedia. Although DCCP's CCID 3 is recommended for transmission of large streaming media, such as videos, this congestion control mode is modelled on the behaviour of the TCP protocol. Thus, properties of the DCCP working in mode 3 are close to the TCP rather than to the properties of typical real-time transport protocols, such as the UDP or RTP/UDP protocol suite. The use of the linear throughput equation, proposed by the Authors in their previous work, models the DCCP's congestion control on the behaviour of the RTP working on top of the UDP.

As was shown in the paper, the linear equation deals with two problematic issues, which are the protocol's equality towards competing flows and the resulting bandwidth constraints imposed by congestion control mechanism. Real-time transmission is rate-sensitive and equality enforced at the level of the transport protocol (instead of, for example, the sending application or codec) and significantly and needlessly reduces the bit rate of the transmitted stream. Experiments shows that our solution of light-weight congestion control for the DCCP, implemented in the Linux kernel, allows the DCCP to send video data in a real-time regime and kept the sending stream congestion controlled. Throughput of the video steam was reduced by no more than 5.5% when competing with 10 TCP flows, and the same reduction of throughput was observed for only 1 stream transmitted by the DCCP working in standard CCID 3 mode. As a result, the proposed mechanism is aggressive enough to compete for bandwidth with TCP flows and not so aggressive as to cause the collapse of the competing TCP traffic.

Acknowledgment. The work was supported by the contract 11.11.230.018.

References

1. Chodorek, A., Chodorek, R.R.: Streaming video over TFRC with linear throughput equation. Adv. Electron. Telecommun. **1**(2), 26–29 (2010)
2. Chodorek, R.R., Chodorek, A.: ECN-capable TCP-friendly layered multicast multimedia delivery. In: Proceedings of UKSim 2009: 11th International Conference on Computer Modelling and Simulation, 25–27 March 2009, Cambridge, England, pp. 553–558 (2009)
3. Chodorek, R.R., Chodorek, A.: Expanding the Ns-2 emulation environment with the use of flexible mapping. In: Gaj, P., Kwiecień, A., Stera, P. (eds.) CN 2016. CCIS, vol. 608, pp. 22–31. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39207-3_2
4. Fall, K., Vradhan, K.: The ns Manual (2014). http://www.isi.edu/nsnam/doc/ns_doc.pdf
5. Floyd, S., Kohler, E.: Profile for datagram congestion control protocol (DCCP) congestion control ID 2: TCP-like congestion control. RFC 4341 (2006)
6. Floyd, S., Kohler, E., Padhye, J.: Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 3: TCP-Friendly Rate Control (TFRC). RFC 4342 (2006)
7. Floyd, S., Handley, M., Widmer, J.: TCP Friendly Rate Control (TFRC): Protocol Specification. IETF RFC 5348 (2008)
8. Floyd, S., Kohler, E.: Profile for datagram congestion control protocol (DCCP) congestion ID 4: TCP-friendly rate control for small packets (TFRC-SP). RFC 5622 (2009)
9. Fujikawa, T., Takishima, Y., Ujikawa, H., Ogura, K., Katto, J., Izumikawa, H.: A hybrid TCP-friendly rate control for multimedia streaming. In: Proceedings of 18th International Packet Video Workshop (PV), pp. 134–141 (2010)
10. Holmer, S., et al.: A Google Congestion Control Algorithm for Real-Time Communication. Internet-Draft draft-alvestrand-rmcat-congestion-03 (2015)
11. iPerf - The TCP, UDP and SCTP network bandwidth measurement tool. https://iperf.fr/. Accessed Mar 2018
12. Parameter values for the HDTV standards for production and international programme exchange. Recommendation ITU-R BT.709-6 06/2015 (2015)
13. Kohler, E., Handley, M., Floyd, S.: Datagram Congestion Control Protocol (DCCP). RFC 4340 (2006)
14. Mahrenholz, D., Svilen, I.: Real-time network emulation with ns-2. In: Proceedings of the 8th IEEE International Symposium on Distributed Simulation and Real Time Applications, Budapest, Hungary, pp. 29–36 (2004)
15. Mahrenholz, D., Ivanov, S.: Adjusting the ns-2 emulation mode to a live network. In: Müller, P., Gotzhein, R., Schmitt, J.B. (eds.) Kommunikation in Verteilten Systemen (KiVS). Informatik aktuell, pp. 205–217. Springer, Heidelberg (2005). https://doi.org/10.1007/3-540-27301-8_17
16. Shiang, H.P., van der Schaar, M.: Quality-centric TCP-friendly congestion control for multimedia transmission. IEEE Trans. Multimedia **14**(3), 896–909 (2012)
17. Singhal, N., Sharma, R.M.: Survey on TCP friendly congestion control for unicast and multicast traffic. Int. J. Comput. Appl. **26**(1), 23–30 (2011)
18. Stevens, W., Allman, M., Paxson, S.: TCP congestion control. RFC 2581 (1999)
19. VQEG: VQEG HDTV TIA Source Test Sequences. ftp://vqeg.its.bldrdoc.gov/HDTV/NTIA_source/. Accessed Mar 2018
20. VideoLAN - Official page for VLC media player, the Open Source video framework! http://www.videolan.org/vlc/. Accessed Mar 2018

On Improving Communication System Performance in Some Virtual Private Networks

Tomasz Malinowski(✉) and Jan Chudzikiewicz(✉)

Faculty of Cybernetics, Military University of Technology,
ul. Gen. Witolda Urbanowicza 2, 00-908 Warszawa, Poland
{tomasz.malinowski,jan.chudzikiewicz}@wat.edu.pl
http://www.wat.edu.pl/

Abstract. The paper presents the procedure for determining the optimal set of hubs and spokes and thus the collection of tunnels in hub-and-spoke network. The subject of the study was a multi-departmental network, with security of transmission requirement, so DMVPN technology with IPSec-protected tunnels were used. The optimization of the hub/spoke set was intended to improve inter-departmental communication efficiency, which was to be confirmed by the analysis of simulation results. In the simulation studies, the quality of the chosen service (VoIP) was checked for the different structures of tunnel connections. Simulation tests were prepared and implemented in Riverbed Modeler environment.

Keywords: Optimal communication structure · Routing protocols
Dynamic tunneling in VPN

1 Introduction

This article is a continuation of the considerations on improving the reliability and performance of transmission in the networks based on hub-and-spoke logical architecture. The procedure for monitoring and maintaining a network of dynamic VPN tunnels was introduced in [1]. The idea of reconfiguration was intended to use a diagnostic theory to test the interoperability of branch boundary routers, which act as a tunnel brokers for other routers, to communicate branches networks using dynamic IPSec tunnels. The basis for this discussion was Cisco DMVPN technology.

The authors of this article focus on the method of centralized monitoring of communication parameters in DMVPN network with scattered departments and on calculation of the optimal set of hubs and spokes. Network structures in which inter-departmental communication takes place through hub routers and is secured by IPSec are considered. The task is to determine optimal set of border routers (single router can act as hub or spoke) and thus a minimal set of VPN tunnels.

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 64–73, 2018.
https://doi.org/10.1007/978-3-319-92459-5_6

This task belongs to the wide domain of determining the optimal allocation of resources and determining the optimal communication structure and is raised in many research works focused on improving reliability, performance, and usability of transmission systems (multiprocessor systems, military computer networks - wired and wireless networks of stationary or mobile nodes). A lot of research focuses on the problems of optimization of the hypercube structure and on the problems with location and relocation of network resources ([2–8]). In the era of IoT (Internet of Things), results of research on optimal and "energy-efficient" communication structures, prolonging the life time of the network with battery-powered nodes, are particularly important [9]. The research is focused on developing new routing protocols, efficient medium access protocols, selecting of nodes collecting and processing information from other nodes (sink nodes) and on indicating nodes of the meeting points (rendezvous points) or nodes acting as coordinators ([9–12]).

In our area of interest, i.e. dynamic tunneling in VPNs, research is focused on testing of the network performance in cases of utilization of various routing protocols [13].

In this paper we introduce a procedure for determining the optimal set of hubs and spokes in specific communication structure. We assume arbitrarily that the dynamic routing protocol, used for broadcasting of prefixes of branch network addresses, is OSPF (Open Shortest Path First). The result of the procedure is the graph of logical VPN connections between departmental border routers. Based on the graph we will be able to distinguish spoke and hub/spoke routers. We assumed that such structure of VPN tunnels will enable efficient and secure exchange of information between branches. We were looking for confirmation of assumptions in the results of the simulation studies.

This paper firstly presents short description and basic DMVPN problems. In Sect. 3, the structure of the analyzed network, the assumptions and formal description of the problem are given. Section 3 also deals with the procedure for determining the best tunneling structure between the border routers of our network. The last section shows the model of simulated network and results of simulation tests.

2 DMVPN Characteristics and Main Transmission Issues

DMVPN (Dynamic multipoint VPN) is a great technology to create scalable VPN. It is based on Multipoint Generic Routing Encapsulation protocol (mGRE), IPSec and Next Hop Resolution Protocol (NHRP). It makes possible build a communication structure connecting branches of the company through secure tunnels over WAN. Some of the tunnels are permanent, some of them are dynamically built-up, as needed, so using DMVPN makes it unnecessary to create and to manage large numbers of tunnel connections in big multi-site networks.

DMVPN consists border routers that function as hubs or spokes. In basic DMVPN form, inter-branch communication via hub nodes is offered (DMVPN

Phase-1) and is called spoke-hub-spoke communication. Another variant is the communication between boundary branch routers through dynamic tunnels, which are established after reconciling of tunnel parameters with the tunnel broker. Spoke is the reconciliation node, and hub is tunnel broker (DMVPN Phase-2, where branches are directly connected). In both cases it is possible to protect the tunnels by IPSec. Implementation of direct communication between branches with secured tunnels is DMVPN Phase-3 (spoke-to-spoke connections with IPSec).

The general shape of DMVPN was shown in Fig. 1.

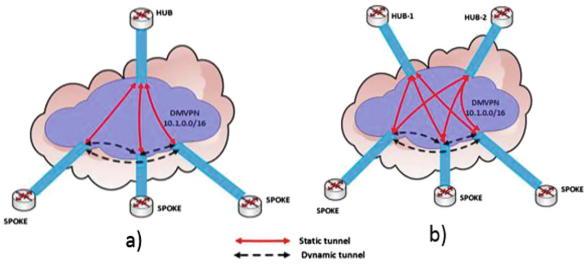


Fig. 1. DMVP with one HUB (a) and backup NHS server (b)

For reliability DMVPN can contain primary and backup hub (structure b on Fig. 1). Regardless of DMVPN structure, each spoke has to register itself in the primary hub or additionally in backup hub, which are called Next Hop Servers (NHS).

DMVPN technology has many advantages, but in various studies the shortcomings and problems that occur in networks with this solution are signaled. For example, for spoke-to-spoke communication (DMVPN Phase-3) a hub failure can directly impact spoke-to-spoke connectivity in those DMVPN networks where spokes can't establish direct IPSec sessions (due to NAT or other limitations).

Another example might be the problem discussed in the article [1]. The authors point out the fact, that the primary condition of successful tunneling is proper functioning of the spoke in hub registration mechanism, the polling mechanism about physical addresses of the final points of tunnels, but also mechanism of call's disconnecting in case of hub failure (connection with hub can be temporary loss, hub can be overloaded, hub's response time can be too long, etc.). In some cases of tunnel connections protected by IPSec, it is necessary to track hub's availability and to remove IPSec session after a period of temporary hub's unavailability (cleaning of IPSec Security Associations) [1].

Despite the fact that DMVPN technology is developed and improved over the years, it is still possible to indicate the situations in which network management procedures in situations of malfunction are useful.

3 Determining of Optimal VPN Communication Structure

Our considerations will be focused on the VPN and the network structure graph G_1, shown in the Fig. 2.

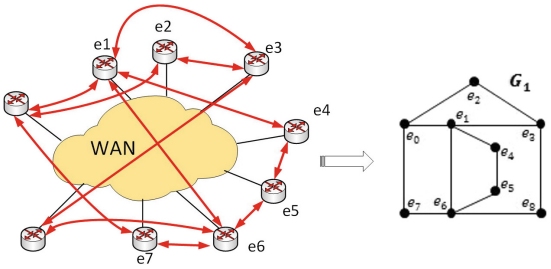


Fig. 2. Structure of potential VPN connections in a specific network

We assume that the analyzed structure of tunnel connections results from some technical limitations of border routers, which means that not every router can act as a hub. All routers, however, can be a spoke. Thus, the initial structure is not the structure of full-mesh tunnel connections but partial-mesh. Such a structure will be a subject of optimization, because we want to minimize the number of the tunnel connections.

The procedure for determining the optimal structure will be implemented centrally by the network management station. The assignation of such structure will involve uploading of the appropriate configuration to the border routers of DMVPN branches.

The network structure is described by graph $G = <E, U>$, for E – set of network nodes, and U – set of communication links (corresponding to DMVPN tunnels).

In the example network, nodes e_0 to e_8 are departmental border routers. Red lines (in Fig. 2) between nodes are potential IPSec tunnels. Graph G_1 shows VPN connections in a simpler form.

VPN topology in Fig. 2 are redundant connections structure, but not full-meshed. Graph G_1 is a subgraph of four-dimensional hypercube It is a redundant structure with high degree of redundancy, fulfilling the stringent requirements imposed on some networks. The issue of determining such subgraphs was discussed in [7].

Our first goal was to implement a procedure of acquiring by management station information about the quality of the transmission channels between nodes $e_0 - e_8$. Because Cisco routers are the border routers of our network, we use IP SLA probes. IP SLA is a great tool and is an embedded agent in Cisco IOS, designed to measure and monitor common network performance metrics like

jitter, delay, and packet loss. IP SLA has two components: the source and the target. The source generates packets, and the target functions as responder.

Simple IP SLA probe may look as follows:

ip sla 1
udp-jitter 10.0.0.1 codec g729a
frequency 40
ip sla schedule 5 life forever start-time now

On the recipient's side, we just turn on the responder using the command: *ip sla responder*

Statistics given by the probe is listed below (some lines are omitted).

WA#show ip sla statistics

Latest RTT: *192 milliseconds*
Source to Destination Latency *Min/Avg/Max: 2/75/244 milliseconds*
Destination to Source Latency *Min/Avg/Max: 2/214/423 milliseconds*
Source *to Destination Jitter Min/Avg/Max: 0/13/209 milliseconds*
Destination *to Source Jitter Min/Avg/Max: 0/23/314 milliseconds*
Packet Loss Values:
Loss Source to Destination: 0
Loss Destination to Source: 0
Out Of Sequence: 0 Tail Drop: 19
Packet Late Arrival: 0 Packet Skipped: 1
Voice Score Values:
Calculated Planning Impairment Factor (ICPIF): 14
MOS score: 4.00

We assume that the probes will be running on border routers, that can set up tunnels as in Fig. 2. The probes will assess the "quality of the tunnel connection". This parameter will be the key to determining the best communication structure of our network.

Because it is communication structure with permanent tunnels (DMVPN Phase-1), our intent is to choose "optimal minimum set" of hubs and spokes (in our case, providing the best VoIP service performance).

In general, let $d\,(e, e'\,|G\,)$ be the distance between nodes e and e' in a coherent graph G. This is the length of the shortest chain (in the graph G) connecting node e with the node e'.

Nodes of the structure G are characterized by a radius. Let $r\,(e\,|G\,) = \overset{max}{_{e' \in E(G)}}$ $d\,((e, e')\,|G\,)$ be the radius of a node. In the Table 1 radius for all nodes of G_1 were presented.

A basis for determining the communication structure is a dendrite, which provides the minimum number of communication lines.

Let $T = <E, U^*>$ be the dendrite i.e. such coherent acyclic partial graph of G that:

$$\exists\,\langle e',\ e''\rangle \in U \Rightarrow \langle e',\ e''\rangle \in U^* \Leftrightarrow [(d\,(e^*,\ e') \neq d\,(e^*,\ e'')) \wedge d\,(e',\ e'') = 1]$$

Table 1. The radius values for G_1

$e \in E(G_1)$	$r(e\,\vert G_1)$
e_0	3
e_1	2
e_2	4
e_3	3
e_4	3
e_5	4
e_6	3
e_7	4
e_8	4

for

$$r(e^*) = \underset{e \in E(G)}{max}\, r(e)$$

The algorithm for dendrite T determining was presented in [7]. The procedure described there gives, in the first step, a set of dendrites, illustrated in the Fig. 5.

For simplicity of calculation, it was assumed that distance between e and e', which are connected via VPN tunnel, is 1, but you can use all or some of the characteristics returned by the IP SLA probes and assign a metric according to your preferences. In a real network environment, we assume continuous (periodic) checking of various SLA parameters, and using them as a metric of inter-nodes tunnels.

The optimal structure is a dendrite determined for a node with minimal radius. For the structure G_1 the node e_1 (Table 1) has a minimal radius. The structure T_{OPT}, shown in Fig. 3, was chosen as the optimal communication structure for the G_1. Thus, the nodes e_2, e_5, e_6, e_7, e_8 are spokes, and the nodes e_0, e_1, e_3, e_4 are hubs.

After determining the optimal communication structure, the management station can perform appropriate reconfiguration of the boundary routers.

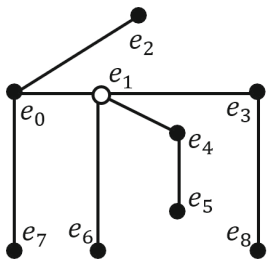


Fig. 3. The optimal communication structure T_{OPT} for the G_1

4 The Results of Simulation Studies

The procedure for determining the set of hubs, spokes and tunnels was verified by simulation tests. The study was conducted in the Riverbed Modeler simulation environment. The simulation model of our network is shown in the Fig. 4.

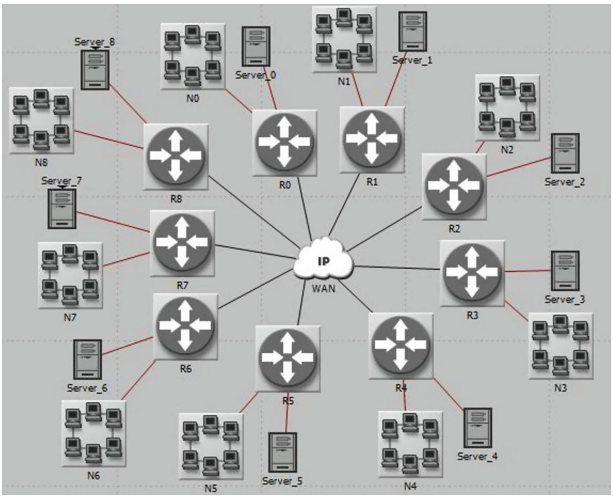


Fig. 4. Simulation model of DMVPN

There were nine branches attached to the WAN by border routers R0 – R8. The single branch was modeled as a LAN with 10 workstations and one http server. Branch workstations can communicate through WAN over VoIP and http (treated as background traffic). The observed service, as especially important for us, was VoIP.

The conducted simulation tests certainly do not serve to verify the design of any network. The tests were to authenticate the procedure for determining the optimal VPN communication structure. Therefore, we only took care of the homogeneity of links and network devices. What was important for us is that in the randomness of generating network streams and the randomness of client-server connections, the simulation results confirm the correctness of the theoretical arguments.

We were interested in the value of important VoIP service parameters like end-to-end delay and delay variation Network services have used the standard application models, available at Riverbed Modeler ("Heavy Browsing" http model and "PCM Quality Speech" voice model). Routers were connected to WAN over T1 links.

Simulation scenarios corresponded to the structures from Fig. 5. It was expected that the results of the simulations will confirm the choice of optimal hubs, spokes and tunnels collections (Fig. 2). Some interesting results Voice

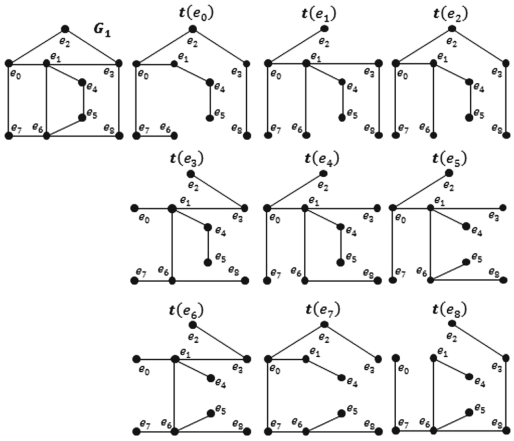


Fig. 5. Set of G_1's dendrites

End-to-End Delay and Voice Delay Variation, confirming the correctness of the procedure for determining the optimal VPN structure are shown in the figures below. *End-to-End Delay* is "average delay in seconds for all network nodes communicating each other under VoIP" and Voice Delay Variation is "average variance in seconds (for all voice workstations) among end to end delays for voice packets. It is measured from the time packet is created to the time it is received" [14]. Low values of these parameters are desirable.

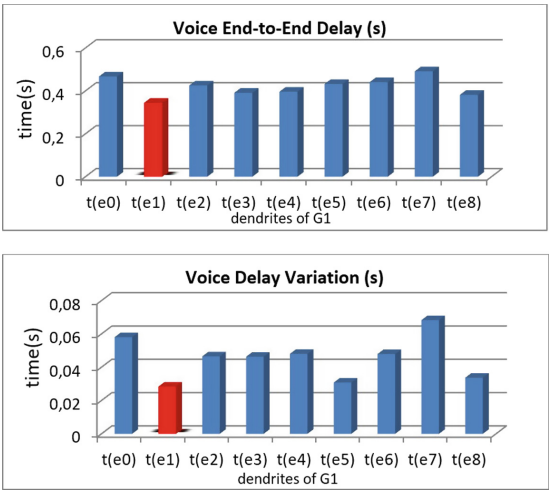


Fig. 6. Average values of *Voice End-to-End Delay* and *Delay Variation*

Complete set of results, in the form of a bar chart, for all G_1's dendrites are presented in Fig. 6. The highlighted bar refers to the best structure $t(e_1)$ (T_{OPT} from Fig. 3).

5 Conclusions and Future Work

Correctness of developed method and its usefulness was confirmed by simulation results. We are satisfied with the simulation results, as considered structures were rather simple and very similar to each other and we did not expect such unequivocal results, which would confirm the correctness of the analytical procedure for determining the optimal VPN structure.

Our next step is the practical implementation of the VPN connection management system. Despite the confirmation of usefulness of our procedure in a simulation environment, it will be interesting to implement and test the effects of continuous monitoring of VPN transmission channels and changing the configuration of routers in a real network environment.

References

1. Malinowski, T., Arciuch, A.: The procedure for monitoring and maintaining a network of distributed resources. In: Annals of Computer Science and Information Systems, Proceedings of the 2014 Federated Conference on Computer Science and Information Systems, vol. 2, 7–10 September 2014, Warsaw, Poland (2014)
2. AlBdaiwia, B.F., Bose, B.: On resource placements in 3D tori. J. Parallel Distrib. Comput. **63**, 838–845 (2003)
3. AlBdaiwia, B.F., Bose, B.: Quasi-perfect resource placements for two-dimensional toroidal networks. J. Parallel Distrib. Comput. **65**, 815–831 (2005)
4. Bae, M.M., Bose, B.: Resource placement in torus-based networks. IEEE Trans. Comput. **46**(10), 1083–1092 (1997)
5. Imani, N., Sarbazi-Azad, H., Zomaya, A.Y.: Resource placement in Cartesian product of networks. J. Parallel Distrib. Comput. **70**, 481–495 (2010)
6. Moinzadeh, P., Sarbazi-Azad, H., Yazdani, N.: Resource placement in cube-connected cycles. In: The International Symposium on Parallel Architectures, Algorithms, and Networks, pp. 83–89. IEEE Computer Society (2008)
7. Chudzikiewicz, J., Zieliński, Z.: On some resources placement schemes in the 4-dimensional soft degradable hypercube processors network. In: Zamojski, W., Mazurkiewicz, J., Sugier, J., Walkowiak, T., Kacprzyk, J. (eds.) Proceedings of the Ninth International Conference on Dependability and Complex Systems DepCoS-RELCOMEX. June 30 – July 4, 2014, Brunów, Poland. AISC, vol. 286, pp. 133–143. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07013-1_13
8. Chudzikiewicz, J., Malinowski, T., Zieliński, Z.: The method for optimal server placement in the hypercube networks. In: Proceedings of the 2015 Federated Conference on Computer Science and Information Systems. ACSIS, vol. 2, pp. 947–954 (2015). https://doi.org/10.15439/2014F159
9. Brindha, L., Muthaiah, U.: Energy efficient mobile sink path selection using a cluster based approach in WSNs. Int. J. Innovative Res. Comput. Commun. Eng. **3**(3) (2015)

10. Erzin, A.I., Plotnikov, R.V.: Using VNS for the optimal synthesis of the communication tree in wireless sensor networks. In: Electronic Notes in Discrete Mathematics, vol. 47. Elsevier (2015)
11. Ghotra, A., Soni, N.: Performance evaluation of ant colony optimization based rendezvous leach using for mobile sink based WSNs. Int. J. Eng. Res. Dev. **11**(07) (2015)
12. Baby, S., Soman, M.: Rendezvous based techniques for energy conservation in wireless sensor networks - a survey. J. Netw. Commun. Emerg. Technol. (JNCET), **3**(3) (2015)
13. Bahnasee, A., Kamoun, N.E.: Study and analysis of a dynamic routing protocols' scalability over a dynamic multi-point virtual private network. Int. J. Comput. Appl. (0975-8887), **123**(2) (2015)
14. Sethi, A.S., Hnatyshin, V.Y.: The Practical OPNET User Guide for Computer Network Simulation. Chapman and Hall/CRC (2012)

New SDN-Oriented Authentication and Access Control Mechanism

Fahad Nife[1,2(✉)] and Zbigniew Kotulski[1]

[1] Faculty of Electronics and Information Technology, Warsaw University of Technology, Warsaw, Poland
fahadnaim114@gmail.com, zkotulsk@tele.pw.edu.pl
[2] Faculty of Science, Al-Muthanna University, Muthanna, Iraq

Abstract. Software-Defined Network (SDN) is recognized as one of the most important future networking area. SDN architecture is a revolutionary new idea that, moving the traditional network to be software-based, provides more flexibility, high degree of automation and shorter provision time. SDN architecture dynamically separates the control plane from the data (forwarding) plane of the network, which provides centralized view of the entire network and makes it easier for managing and for monitoring the network's resources. However, the initial design of the SDN, with its centralized point of control, does not consider sufficiently the security requirements, which makes the security issues additional challenges. In this paper we propose a new access control system for the SDN architecture, working as a controller application used to verify the identity of a host upon connection to the network. The proposed mechanism, which denies the access attempts from unauthorized hosts and defines different levels of privileges for each host, according to its authentication credentials, is implemented using a POX controller. Our approach neither needs a support of new protocols, nor requires additional configuration of hosts or routers.

Keywords: Software-Defined Networking · IEEE 802.1x Port-based authentication · Network security · Radius

1 Introduction

Software-Defined Networking (SDN) is a new framework which comes to facilitate the network management and to improve the overall network's performance and monitoring by physical decoupling the packets forwarding process of the network (considered as the Data Plane, DP) from the routing process (performed in the Control Plane, CP) [1]. SDN centralizes the network's intelligence in one component called Controller, which has a comprehensive view of the entire network and which is responsible for taking the forwarding decisions for the DP (switches and routers) that just performs basic packet forwarding [2]. To provide more eligibility and scalability to the network, the network applications that provide

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 74–88, 2018.
https://doi.org/10.1007/978-3-319-92459-5_7

network's services are abstracted into a separate plane (Application Plane, AP). In SDN, the centralized controller (representing the network's operating system), the switches (network resources) and the applications (network consumers) can communicate in real-time [3] through the NorthBound Interface (NBI) and the SouthBound Interface (SBI). The Application Plane utilizes the NBI, which is a programmable API, to communicate with the Controller. Typically, there are no standard Northbound Interfaces defined, and the applications should provide their own APIs. The Controller uses the Southbound Interface (e.g. OpenFlow) to communicate with the underlying Data Plane. OpenFlow is a standard protocol (standardized by the Open Networking Foundation (ONF) [4]), widely used for the Southbound interface in SDN Networks [5]. It is typically used for the communication between the Controller and the switches under its control. OpenFlow can provide encrypted communications between the Controller and the underlying infrastructure, using Transport Layer Security (TLS), which is considered as an optional feature, and unprotected communications using clear Transmission Control Protocol (TCP).

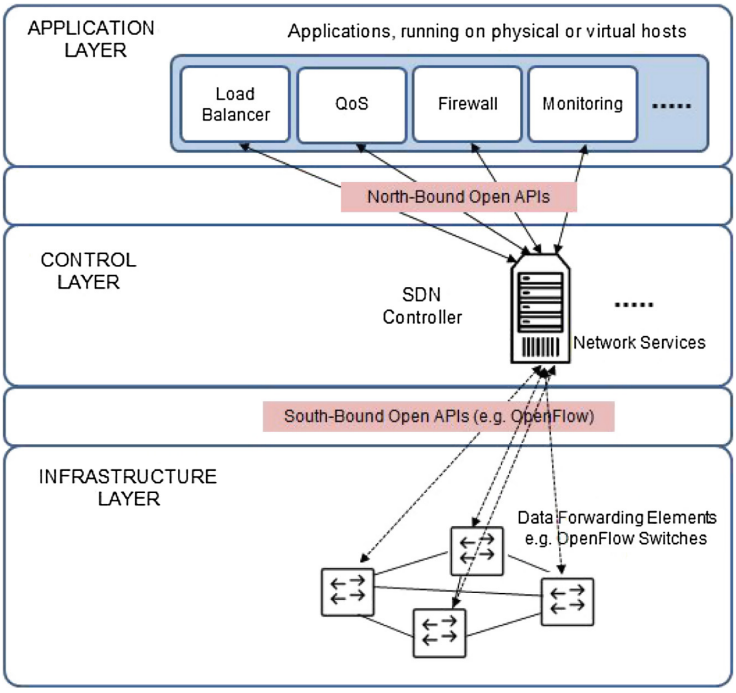


Fig. 1. Software-Defined Network abstraction.

Such centralized intelligence and programmability of SDN give an advantage of providing the Controller with a global view of the entire network, as well

as deploying additional network applications to perform some important services (such as: firewalls, intrusion detection engines and QoS), instead of using proprietary devices [6]. However, in addition to the security threats and vulnerabilities known in the traditional network, this centralized intelligence and direct programmability have their own drawbacks concerning security, scalability, and elasticity. Moreover, the original design of an SDN network is lacking any built-in protection instrument or even basic security mechanism (e.g. authentication and authorization of entities), what has become more and more problematic [7]. Nevertheless, securing the SDN network is very important and is a key feature of a new network idea success, so we cannot consider a secure network without an access control mechanism applied to prevent unauthenticated clients to gain access to the network, or to avoid unauthorized users getting access to sensitive resources of the network [8]. In this paper, we are taking the advantages of the centralized controller with its global view of the network, the flexibility, and the programmability of SDN, to propose a new Network Access Control System and a Policy Enforcement Solution for SDN networks, which will work as an application based on API provided by the Controller, see Fig. 1. Our proposal is based on IEEE 802.1x, i.e., Extensible Authentication Protocol (EAP) over LAN encapsulation and a RADIUS authentication protocol. The Authenticator is centralized and maintains a database of currently active users, as an introduced solution for the stateless property of RADIUS, by keeping records of currently authenticated users to limit concurrent sessions. Such a centralized Authenticator and Controller architecture could easily help in providing authenticated-client mobility inside the network by relating a user/host with its data flows. Since the Controller is the most important entity in the SDN network, which is usually considered as the main goal for attackers, we decided to separate it from, both, the Policy Information Point (PIP), where the users' identities are stored, and the Policy Enforcement Point (PEP), where the policies are enforced (e.g., switch or router). Thus, there is no possibility for any client to reach the Controller. Our proposed solution does not need any additional supporting protocol and does not require new configuration of hosts or routers. The rest of the paper is structured as follows: Sect. 2 presents known authentication solutions in traditional networks, Sect. 3 gives an overview of related works concerning SDN security, Sect. 4 presents the proposed authentication system's design: the new access control method and its data flow policy, Sect. 5 shows the outline of the experiment implementation and the estimated performance of the system, while the last Sect. 6 presents the conclusions and the future work.

2 Network Authentication for Traditional Networks

In today's world, network administrators make a huge effort to secure enterprise networks, which often contain sensitive information and important resources. Regardless of the policies and measures they use, such networks are still vulnerable due to the accidental applications' bugs, user carelessness, or even from insider threats. Access to networks should be simple, however uncontrolled and

unauthorized access is generally not attractive. Identifying users and devices connected to a network is a key part of network's security; it is called Network Access Control and sometimes is referred as Port-based access control. There is a standard technology called IEEE 802.1x [9] which defines a Port-based Network Access Control mechanism for Ethernet networks. IEEE 802.1x is a framework that defines a method for encapsulating EAP (Extensible Authentication Protocol [10]) messages to be sent over a Local Area Network (LAN). This encapsulation is known as EAP over LAN (EAPOL). The idea of such a solution is that before one can access the network through a switch port, he or she must first be authenticated using his or her credentials. So, nobody can directly enter the network. For authentication purposes, IEEE 802.1x relies on EAP protocol, which allows using different authentication procedures like EAP-MD5, EAP-IKE v2, EAP-TLS, and many more. The standard specification of 802.1x includes three major components, which are: the Supplicant (host), the Authenticator, and the Authentication Server, see Fig. 2.

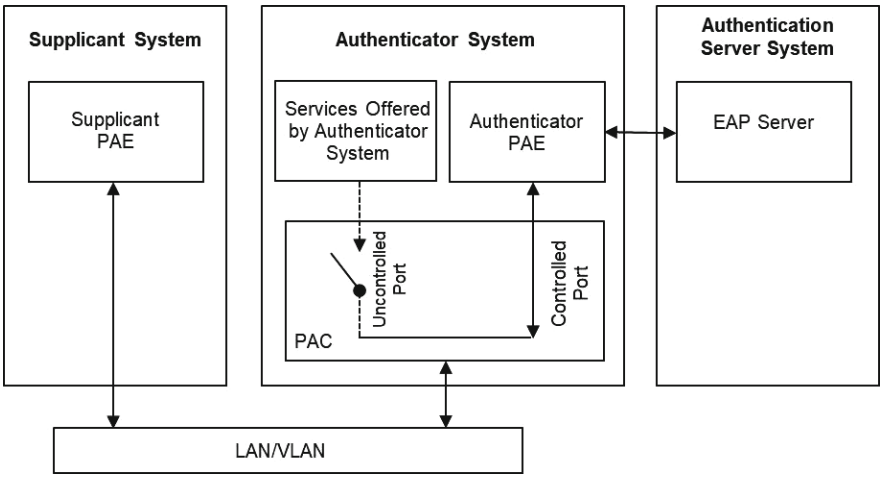


Fig. 2. IEEE 802.1x framework.

The Supplicant is an agent of the service running on a device that is ready to access the network. It responds to the requests coming from switches. Authenticator acts as a proxy between the Supplicant and the Authentication Server. It controls the access to the network based on the authentication status of the Supplicant. It requests the Supplicant identity information, verifies this information with the Authentication Server, and releases the authentication response to the Supplicant. The Authenticator performs the RADIUS client functionality, encapsulates and decapsulates the EAP frames and interacts with the Authentication RADIUS server, see [11]. The Authenticator's physical port can be logically defined either as a controlled port (which permits full access to the network

resources) or as an uncontrolled port (which provides access only for EAPOL traffic between the Supplicant and the Authentication Server), which give two virtual access points. Before the Supplicant is authenticated, the uncontrolled port is the only open port in the Authenticator, but after the Supplicant is successfully authenticated, the controlled port is opened. The third major component of the framework is the Authentication Server, which performs the actual authentication to the Supplicant. The Authentication Server is, generally, a host running software supporting the RADIUS and the EAP protocols. Its role is to validate the identity of the Client and to notify the Authenticator, if the Client is authorized to access the network or not. All authentication and authorization policies are stored on the Authentication Server and, depending on the access level of the user, the specific policies are applied. The Authentication Server and the Authenticator utilize such protocols as RADIUS (Remote Authentication Dial in User Service) protocol [11] or Diameter protocol [12] to transport and exchange EAP frames.

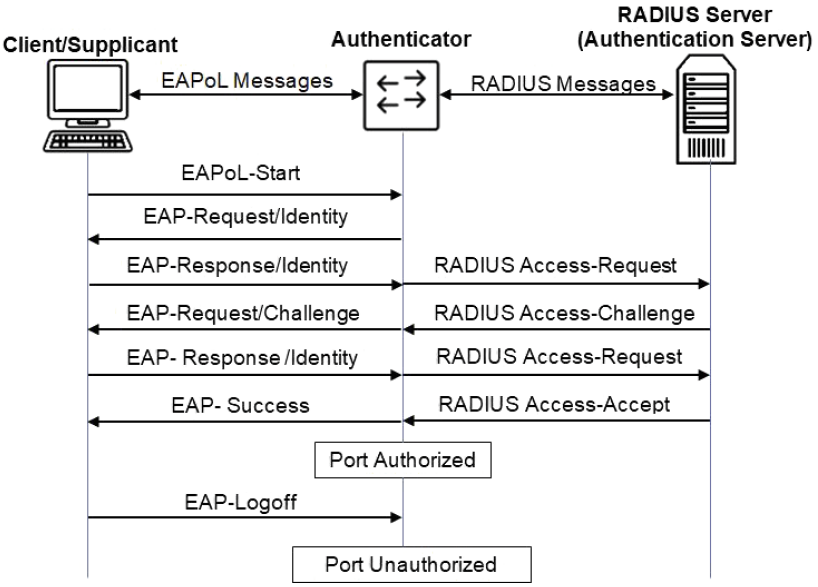


Fig. 3. Messages exchanged during authentication.

The steps of the authentication procedure are the following. The Supplicant initiates the communication when it detects a change of the link state from down to up, while the port remains in the unauthorized mode during the authentication process. The Authenticator sends EAP-Request/Identity frame to the Supplicant, asking for its identity or the credentials. Once the Supplicant receives the request, it responds with the EAP-Response/Identity frame containing the credentials. Now, the Authenticator begins its role as a proxy between the Supplicant and the Authentication Server; it passes the EAP frames between them

until the validation is successful or it fails. If the authentication is successful, the port is authorized and then the user can access the network. In Fig. 3 are shown the messages exchanged between the components. It is a little bit complicated process, but it ensures that only the permitted clients can access the network's resources.

3 SDN-Dedicated Network Authentication Systems

Meeting the need of protecting the SDN networks, recently different dedicated security solutions have been presented, for instance, IDS [13], IPS [14], Stateless Firewall [15], Stateful Firewall [16], reactive Firewall [17], and Distributed Firewall [18]. In this section, we give an overview of several known Network Access Control Systems for SDN networks and then compare them with our proposed solution. Thus, the Ethan project [19] relies on defining a global policy for the centralized controller architecture and a strong binding between the users' attributes, their machines and their traffic (IP, MAC, and Ports) to provide network access control system based on user's identity. Ethane argues that all users, hosts, and switches must be registered and authenticated before any communication. The hosts are authenticated by their registered MAC addresses, the users are authenticated by presenting their usernames and passwords to a website end of the Kerberos server, and the switches use certificates to authenticate themselves to the controller through an SSL secure channel. Resonance [20] extends the Ethane to allow the dynamic network policy based on the real-time monitoring. In Resonance, each device connected to switch has, both, the "Security Class" that transcribes, how the traffic flows to the resources, and the "State", that determines which Security Class should be applied at that time. Like in Ethan, the user is redirected to a website to submit his credentials to be authenticated. The solutions presented in [21,22], AuthFlow [23], FlowNAC [24], and FlowIdentity [25] attempt to provide authentication and access control systems for SDN networks using IEEE 802.1x with the RADIUS Authentication Server, but they differ, basically, with how the host/end user is authenticated, and with how the RADIUS Client is used as the Authenticator. The authors in [21,22] have proposed a Network Access Control System for an OpenFlow-based SDN network, which is compatible with the traditional network infrastructure. In this solution, newly connected hosts are directed to the authentication website with an integrated RADIUS Client to provide their credentials. The website is ending to RADIUS Authentication Server. Similarly to Ethane [19], their way of host authentication is based on tight binding the host to the port on the switch. However, the captive portal is the common background in these proposals; it requires Layer 3 configuration (e.g. DHCP) or even ephemeral private IP address to access the website. Moreover, these approaches require an HTTPS-capable Web browser installed in the host to be able to authenticate, which is a problem for virtual machines or some network devices, like printers, scanners, phones, or even some servers that do not support the web browser. In contrast, our proposed solution does not utilize a website; instead, it uses a hostapd [26] as

Authenticator and adopts the standard 802.1x that does not need an IP address assignment before the authentication process. AuthFlow [23] authenticates hosts directly at Layer 2 (based on MAC addresses), then it provides different privileges for users, based on their credentials. AuthFlow extends the RADIUS Client authenticator hostapd to communicate with the controller through an SSL-based secure channel. AuthFlow differs with our proposed system; it only provides the Access Control system for SDN networks, while our solution provides the Access Control system and introduces a mechanism for the stateless property limitation of RADIUS protocol, by enforcing the Authenticator to maintain a local database of the authenticated active users. FlowNAC [24] proposes a Flow-Based Network Access Control for SDN networks, in which the flows are associated to services; the services are uniquely identified, and the decisions are based on, both, the user's identity and the requested services. FlowNAC enforces the extended centralized policy (e.g., target service identifier) to be implemented in a proactive manner. FlowNAC implement IEEE 802.1x standard, however, to provide the user with the ability to access multiple services simultaneously, the EAPoL_in_EAPoL encapsulation has been introduced, which in turn needs the extension and updating the protocols, entities' exchanged information, and data models. FlowIdentity [25] present an SDN authentication solution using IEEE 802.1x standard with authorization (policy enforcement) through a role-based firewall. This proposal does not change the traditional 802.1x port-based authentication; however, the defined policy can be dynamically updated and directly enforced. Both, FlowNAC and FlowIdentity, argue to provide a service-based authentication, which differs only in the way that the policy enforcement is achieved. FlowNAC and FlowIdentity are differing from our solution; they are implementing the service-based authentication. FlowIdentity differentiates the service using the role-based firewall, while FlowNAC employs EAPoL_in_EAPoL encapsulation and utilizes the predefined rules as a set of rules per service. In contrast, our proposed solution is the identity-based authentication, it does not need any additional supporting protocols or does not require new encapsulation, which might lead to additional unrequired overhead. In the paper [27], the authors have suggested SEaaS (Security as a Service) model, which provides mutual authentication mechanism between an application and the controller in SDN networks. It provides an additional security service for Southbound SDN API (i.e. REST API), which is not supported by an existing protocol. SEaaS is focused on providing the authentication between the controller and the applications that run above it, while our proposed work is identity-based authentication between the controller and client/users who try to access the network. The authors of [28] have implemented an authentication and authorization module, which is a network application maintaining a wide-session database. It has been implemented with two different modes: pass-throw mode and authentication server mode. The main difference between this solution and our proposed work is the place where the Authenticator is located. In [28] the Authenticator has been integrated with the Controller, which may lead to increase the load on the controller and make the solution vulnerable to attacks on the controller, like SYN flooding, which

may finally destruct the whole network. We implement the Authenticator as a separate entity, which helps to isolate the Controller from an unrequired access. The main aim of our proposed solution is to build an Access Control system for securing the SDN network by employing the 802.1x standard with RADIUS server to authenticate, both, the users and hosts, which is based on the Centralized Authenticator separated from the Controller. Moreover, it introduces a solution for the stateless property of the RADIUS protocol by modifying the Centralized Authenticator to maintain a local database used to keep records of currently authenticated users and to limit the concurrent sessions. Every new authentication request is compared against this database before forwarding it to the Authentication Server.

4 System Architecture

4.1 Architecture Overview

The main objective of this work is to design and develop a network access control and authorization mechanism for an SDN network. The proposed architecture's overview is shown in Fig. 4, in which the 802.1x standard is adopted for SDN networks, as it does not enforce any changes on the host sides. The architecture is composed of four components, which are: the OpenFlow-enabled switches, the Controller, the Authenticator, and the Authentication Server. The policy is enforced in a reactive mode (based on the first packet), what keeps the data plane flow table small, instead of filling it with unmatched rules. However, only two entries should be installed in advance in the switch's flow table, to enforce the switch forwarding all EAPOL (Ethernet type set to 0x88E) to the Authenticator, while dropping all other packets. The Centralized Authenticator is the modified version of the hostapd. It is separated from the controller for security reasons (to isolate the controller for protecting the network from presumably the most extreme threats on SDN networks, where such vulnerabilities and attacks on the Controller will lead to compromise the whole network, see [29,30]). The Authenticator is implemented as a RADIUS Client. It receives the EAPOL frames from the switches, decapsulates the EAP frame, compares its contents with a local database of currently active users and, in case of no match, re-encapsulates it in the RADIUS frame and forwards it to the RADIUS Authentication Server. The reply from the Authentication Server, whether it succeeds or fails, is forwarded to the controller via a secure encrypted channel using SSL 3.0 standard established between the Controller and the Authenticator. The Authentication Server (RADIUS) basically performs the actual authentication process by validating the identity of the Client and notifying the Authenticator if the Client is authorized to access the network or not.

4.2 Authentication Process

The authentication process starts when a new 802.1x-aware Client (Supplicant) connects to the network. The authentication messages exchange between

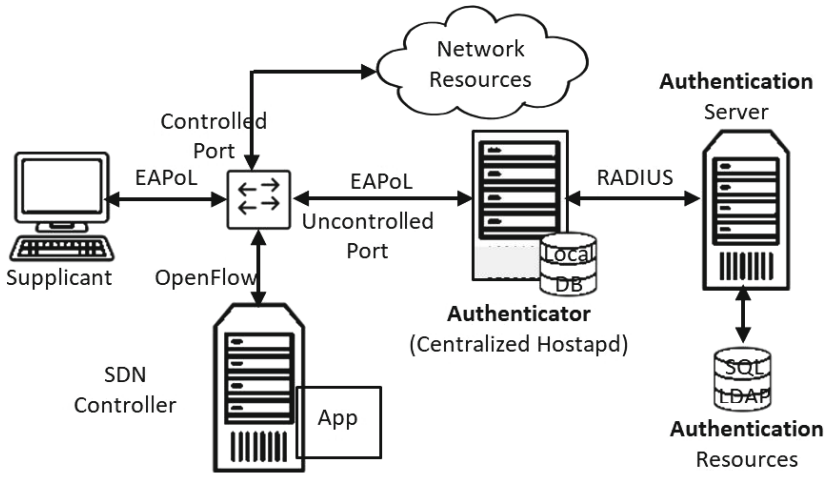


Fig. 4. Proposed architecture.

the Supplicant and the Authenticator starts either when the Supplicant is sending EAPOL-Start frame to the edge switch, or when the Authenticator detects state changes in one of its ports and, alternatively, when it receives a packet on a specific port, with a source MAC address not included in its flow table, see Fig. 5. The switch is proactively instructed to forward EAPOL-start frames to the Centralized Authenticator. The Authenticator responds with EAP-Request/Identity, asking for the Supplicant credentials. The Supplicant answers with EAP-Response/Identity packet providing its credentials (user name that uniquely defines this request for the Client). Then, the Authenticator decapsulates the message and compares its contents against a local database of currently active users. In case the match occurs, the Authenticator informs the Controller application. Otherwise, it will encapsulate the message and will forward it to the Authentication Server. The Authentication Server responds with RADIUS Access/Challenge to be forwarded to the Supplicant. After that, the Supplicant provides its identifying credentials using EAP-Response/Identity. Then the RADIUS server verifies the received credentials and transmits either an EAP-Success (Access-Accept) Packet, if the Client is successfully authenticated and authorized to gain access to the network, or EAP-Reject (Access-Accept) packet what means the access is denied and the port is kept blocked. The Authenticator will update its active-user database and will forward the final decision to the Controller, which accordingly may install new entry on the corresponding switch's flow table and apply the predefined policy group for that Client.

4.3 Authorization Process

Typically, authentication and authorization are coupled together. After successful authentication we know "who can access", while authorization means "who

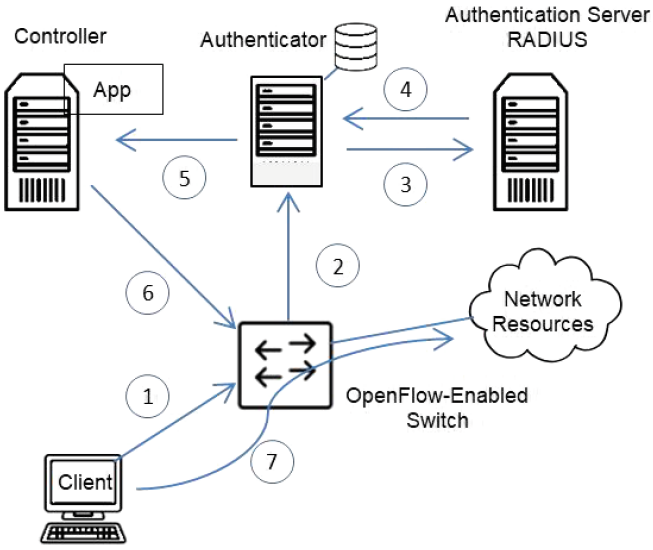


Fig. 5. Authentication process.

can access what" in the network. This procedure concerns users, devices, and network services. So, when the Supplicant is correctly authenticated, the Authentication Server returns an EAP-Success (Access-Accept) Packet, which includes a list of attribute-value pairs (user privileges) that describes the parameters to be used for the current session. Then, the modified hostapd authenticator forwards these messages to the application running on the top of the Controller to inform about the success of authentication and the identity of the Supplicant (its MAC address). Such parameters may include: the service type, the protocol type, the access list to apply, or just the static route to be installed in the OpenFlow table. Finally, the application running over the Controller translates these parameter into flow rules to be installed in the responsible switch and applied on the corresponding port. In such a way, the switch can easily find correlation between the Supplicant and its flow.

4.4 Database

A Centralized Database with dynamic capacity is maintained to provide means for fine-grained network access control system and to keep states of currently authenticated active users, which leads to limiting the concurrent sessions for each authenticated user. This Database is located at the Authenticator side and is used to store information about all currently active users. For each successfully authenticated user, a new entry is added. This Database should be searched first for every authentication request before fetching information from the Authentication Server. The port-down event messages sent from the switch to the Controller are a good sign to remove the corresponding entry from the Database.

5 Implementation and Functional Validation

5.1 Implementation

The Network Access Control solution, proposed in Sect. 4, has been validated at the functional level. A proof-of-concept solution has been implemented and several experiments have been designed and performed. In Fig. 6 we present a draft view of the validation environment that has been used to prove the functional viability of our proposal. We have built up our testbed that includes the following virtual machines:

- Authentication Server: Ubuntu Server 16.04.3 LTS VM to run the FreeRADIUS [31] server v2.1.12., with MySQL database to store the authentication and authorization resources.
- Authenticator: Ubuntu Server 16.04.3 LTS VM to run the modified version of the open-source 802.1x hostapd [26]. It is modified to be able to maintain a local database as well as to establish a secure channel with the Controller.
- Controller: Ubuntu Server 16.04.3 LTS VM to run the POX [32] Controller along with two applications, forwarding.l2_learning module and our own application, which is responsible for installing or removing entries in the switch, based on the results of authentication and authorization processes, collected from the Authenticator, via the SSL channel.
- Switch: Ubuntu Server 16.04.3 LTS VM to run the Mininet [33] network emulator used to create the Open vSwitch [34] switch establishing connections between different entities, for testing purpose.
- Supplicant: Ubuntu Desktop 16.04.3 LTS VM installed wpa-supplicant for Network Clients. In the testbed there are two Supplicants (Client-1 and Client-2), one server (providing different services to the network's devices), the Centralized Authenticator (maintaining a local database of active authenticated clients), the Authentication Server (storing the users' credentials), and the Controller (running the proposed application).

5.2 Validation

To test the correct behavior of our proposed solution and, next, to estimate its effectiveness, different test and experiment scenarios have been designed and (partially) implemented. The first test is to show the efficiency of allowing the authenticated user's traffic while dropping an unauthorized traffic. For this test, Client-1 is correctly authenticated and therefore the Controller installs entries in the switch's OpenFlow table, which explicitly reflect the client's privileges. So, Client-1 can access the network resources, while the other host, Client-2, is an unauthorized host, so its access request is rejected and no its OpenFlow entries installed, so the switch refuses all its attempts to access the network resources. The second experiment is to show "who can access what" in the network. The simple match-action paradigm offered by the OpenFlow protocol can give a great

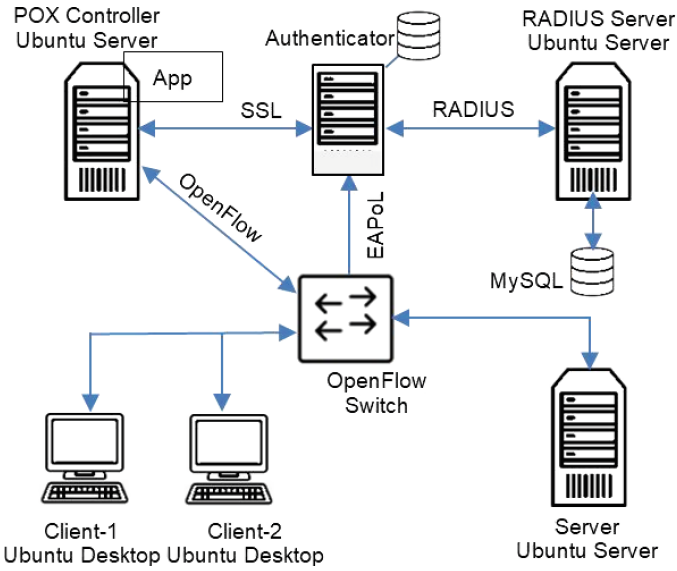


Fig. 6. Experiment testbed.

utility of traffic control by giving an ability to filter packets, based on the protocol, source and/or destination IP addresses, and source and/or destination port numbers. For instance, it blocks an address from the accessing HTTP session of a specific server, while it allows its other services. In fact, using TCP and UDP ports makes it possible to support or suppress higher level applications. Generally, the port numbers are associated with applications, so allowing or denying the access to a specific port number determines, which application can be used, and which devices can access the port. This allows setting up the network to carry only certain types of traffic, for instance, the FTP (File Transfer Protocol) packets or POP3 mail. In this test, the authenticated Client-1 is allowed to start an HTTP Sessions, while it is forbidden to access Telnet sessions. So, some of the entries installed are allowed any request to TCP port (80/8080) and are denied any request to TCP port equal to 23. The last experiment designed is to validate the solution for the stateless property of RADIUS by testing the Authenticator database with two scenarios. In the first scenario, we unlimited the ability to use the same user's credentials, in which, both, Client-1 and Client-2 successfully authenticate themselves using the same credentials. Then (the second scenario), we limit such a possibility to only one session at a time. Therefore, only the first authentication request can be verified successfully. The above tests have been implemented and they confirmed correctness of the protocol, as the protocol worked according to the proposed scenarios. Now, all these tests must be extensively carried out with different environmental constraints to estimate practical performance of the protocol and its resistance to external disturbance. After presentation and discussion of our solution, let us summarize security functions

and specific functionalities of the SDN-dedicated network authentication systems, collecting together basic properties of the known solutions (what has been already presented in Sect. 3). In Table 1 we compare the proposed approach with some systems known from the literature.

Table 1. Comparison of the proposed approach with other SDN-based Access Control solutions

Publication	Authenticator	Layer based	Behavior	Capabilities required
Ethane [19]	Captive portal	Layer 3	Reactive	Web Browser Support
Resonance [20]	Captive portal	Layer 3	Reactive	Web Browser Support
AuthFlow [23]	Captive portal	Layer 3	Reactive	Web Browser Support
FlowNAC [24]	Hostapd	Layer 2	Proactive	EAPoL_in_EAPoL encapsulation
FlowIdentity [25]	Hostapd	Layer 2	Reactive	No
Proposed solution	Hostapd	Layer 2	Reactive	No

6 Conclusions and Future Work

In this paper, we have intended to introduce a new authentication and access control mechanism for OpenFlow-based SDN networks, by adopting the 802.1x standard. The proposed solution provides authentication and authorization mechanisms for SDN networks in such a way, that it neither enforces changes to the SDN paradigm in term of the network's design or behavior, nor it requires any support of new supplementary protocols or even it needs no additional configuration of hosts and routers. Another advantage of our solution is that it intends to overcome the stateless property of the 802.1x protocol by maintaining a centralized active-user database for the fine-grained network access control system, which keeps records of currently authenticated users and which limits the concurrent sessions. In this solution, the security policy is centralized and dynamically enforced to the switches. We have chosen to implement the proposed solution with a reactive mode, to gain an advantage of keeping the data plane's flow tables small. For the future enhancements, we reclaim to implement, both, reactive and proactive modes to improve the overall system's performance. We also plan to deploy our prototype in a real network environment to confirm its functionality and its strength against active attackers.

References

1. Pujolle, G.: Software Networks Virtualization, SDN, 5G and Security. ISTE Ltd. and Wiley, London and New York (2015)
2. Kreutz, D., Ramos, F.M.V., Verissimo, P.E., Rothenberg, C.E., Azodolmolky, S., Uhlig, S.: Software-defined networking: a comprehensive survey. Proc. IEEE **103**, 14–76 (2015)
3. Astuto, B.N., Mendonca, M., Nguyen, X.N., Obraczka, K., Turletti, T.: A survey of software-defined networking: past, present, and future of programmable networks. IEEE Comm. Surv. Tutor. **16**, 1617–1634 (2014)
4. The Open Networking Foundation, OpenFlow Switch Specification (2015). https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf
5. Hu, F., Hao, Q., Bao, K.: A survey on software-defined network and OpenFlow: from concept to implementation. IEEE Commun. Surv. Tutor. **16**(4), 2181–2206 (2014)
6. Lara, A., Kolasani, A., Ramamurthy, B.: Network innovation using OpenFlow: a survey. IEEE Comm. Surv. Tutor. **16**, 493–512 (2014)
7. Ahmad, I., Namal, S., Ylianttila, M., Gurtov, A.: Security in software defined networks: a survey. IEEE Commun. Surv. Tutor. **17**(4), 2317–2346 (2015)
8. Alsmadi, I., Xu, D.: Security of software defined networks: a survey. Comput. Secur. **53**, 79–108 (2015)
9. Local and Metropolitan Area Networks' Port-Based Network Access Control, IEEE Standard 802.1x (2010)
10. Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., Levkowetz, H.: Extensible Authentication Protocol (EAP), RFC 3748 (Proposed Standard) (2004). http://www.ietf.org/rfc/rfc3748.txt
11. Rigney, C., Willens, S., Rubens, A., Simpson, W.: Remote Authentication Dial In User Service (RADIUS), RFC 2865 (Draft Standard) (2000). http://www.ietf.org/rfc/rfc2865.txt
12. Fajardo, V., Arkko, J., Loughney, J., Zorn, G.: Diameter Base Protocol, RFC 6733 (Proposed Standard) (2012). http://www.ietf.org/rfc/rfc6733.txt
13. Jeong, C., Ha, T., Narantuya, J., Lim, H., Kim, J.: scalable network intrusion detection on virtual SDN environment. In: 2014 IEEE 3rd International Conference on Cloud Networking (CloudNet), pp. 264–265, Luxembourg (2014)
14. Francois, J., Aib, I., Boutaba, R.: Firecol: a collaborative protection network for the detection of flooding DDoS attacks. IEEE/ACM Trans. Netw. (TON) **20**, 1828–1841 (2012)
15. Yoon, C., Park, T., Lee, S., Kang, H., Shin, S., Zhang, Z.: Enabling security functions with SDN: a feasibility study. Comput. Netw. **85**(1389–1286), 19–35 (2015)
16. Nife, F., Kotulski, Z.: Multi-level stateful firewall mechanism for software defined networks. In: Gaj, P., Kwiecień, A., Sawicki, M. (eds.) CN 2017. CCIS, vol. 718, pp. 271–286. Springer, Cham (2017)
17. Zerkane, S., Espes, D., Le Parc, P., Cuppens, F.: Software defined networking reactive stateful firewall. In: Hoepman, J.-H., Katzenbeisser, S. (eds.) SEC 2016. IAICT, vol. 471, pp. 119–132. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-33630-5_9
18. Pena, J.G., Yu, W.E.: Development of a distributed firewall using software defined networking technology. In: 4th IEEE International Conference on Information Science and Technology, pp. 449–452, Shenzhen, China (2014)

19. Casado, M., Freedman, M.J., Pettit, J., Luo, J., McKeown, N., Shenker, S.: Ethane: taking control of the enterprise. In: ACM SIGCOMM, Kyoto, Japan, pp. 1–12 (2007)
20. Nayak, A., Reimers, A., Feamster, N., Clark, R.: Resonance: dynamic access control for enterprise networks. In: Workshop: Research on Enterprise Networking (WREN), Barcelona, Spain (2009)
21. Dangovas, V., Kuliesius, F.: SDN-driven authentication and access control system. In: The International Conference on Digital Information, Networking, and Wireless Communications (DINWC). Society of Digital Information and Wireless Communication, pp. 20–23 (2014)
22. Kuliesius, F., Dangovas, V.: SDN enhanced campus network authentication and access control system. In: 2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN), pp. 894–899 (2016)
23. Mattos, D.M.F., Ferraz, L.H.G., Duarte, O.C.M.B.: AuthFlow: authentication and access control mechanism for software defined networking. Ann. Telecommun. **71**(11), 607–615 (2016). https://doi.org/10.1007/s12243-016-0505-z. ISSN 0003–4347
24. Matias, J., Garay, J., Mendiola, A., Toledo, N., Jacob, E.: FlowNAC: flow-based network access control. In: Third European Workshop on Software-Defined Networks (EWSDN), Budapest, Hungary, pp. 79–84 (2014)
25. Yakasai, S.T., Guy, C.G.: Flowidentity: software-defined network access control. In: IEEE Conference on Network Function Virtualization and Software Defined Network, pp. 115–120 (2015)
26. Malinen, J.: Hostapd: IEEE 802.11 AP, IEEE 802.1x/WPA/WPA2/EAP/RADIUS Authenticator. https://w1.fi/hostapd/
27. Green, K., Junghyun, A., Keecheon, K.: A study on authentication mechanism in SEaaS for SDN. In: IMCOM 2017, Beppu, Japan (2017)
28. Hauser, F., Schmidt, M., Menth, M.: Establishing a session database for SDN using 802.1x and multiple authentication resources. In: IEEE ICC 2017 SAC Symposium SDN & NFV Track, pp. 1–7 (2017)
29. Heller, B. Sherwood, R., McKeown, N.: The controller placement problem. In: First Workshop on Hot Topics in Software Defined Networks, ser. HotSDN 2012, pp. 7–12. ACM, New York (2012)
30. Kreutz, D., Ramos, F.M., Verissimo, P.: Towards secure and dependable software-defined networks. In: Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, ser. HotSDN 2013, pp. 55–60. ACM, New York (2013)
31. FreeRADIUS, FreeRADIUS project. https:freeradius.org/
32. POX Controller, POX wiki. https://openflow.stanford.edu/display/ONL/POX+Wiki
33. Mininet: An Instant Virtual Network on your Laptop (or another PC). http://mininet.org
34. O.vSwitch, Open vSwitch - Production Quality, Multilayer Open Virtual Switch. http://openvswitch.org/

Full Network Coverage Monitoring Solutions – The netBaltic System Case

Damian Karpowicz, Tomasz Gierszewski(✉), and Krzysztof Nowicki

Faculty of ETI, Gdańsk University of Technology, Gdańsk, Poland
tomag@pg.edu.pl
https://eti.pg.edu.pl/

Abstract. This paper defines the problem of monitoring a specific network, and more precisely – part of reporting process, which is responsible for the transport of data collected from network devices to station managers. The environment requires additional assumptions, as a specific network related to the netBaltic Project is to be monitored. Two new monitoring methods (EHBMPvU and EHBMPvF) are proposed, which priority is full network coverage. Both are based on employing additional IPv6 headers. Methods have been ana-lyzed in dedicated simulation environment and compared to classic monitoring solution – SNMP. The results show, that one of the proposed solutions outperform the current standard, but depend on traffic characteristics of the network.

Keywords: Network monitoring · IPv6 · netBaltic · Extension headers

1 Introduction

Network monitoring is a term used to describe [1] systematic, continuous control the behavior of the network and all elements included in this network. The process of monitoring network consists of five logical parts [2]: data collection, data representation, reporting, data analysis and presentation of the results. Network monitoring is the key process in the implementation of network management tasks and supports, among other things like network functioning analysis, problem and defects identification and the correctness of network configuration changes verification.

By collecting data more frequently and gathering more and more various types of data, more problems can be identified, but it also increases the network load. This problem becomes more significant with the increasing bandwidth consumption, rising IPv6 protocol popularity and the popularity of the Internet in general [2] due to network traffic growth – and the volume of measurement data grows with it. New system solutions for network monitoring should evolve towards better scalability and performance provision [3], and this is why organizations are still developing better network monitoring systems.

One way to reduce both the consumption of network bandwidth and computational overhead in management stations (centers) is to increase the intelligence in

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 89–102, 2018.
https://doi.org/10.1007/978-3-319-92459-5_8

devices, which gather monitoring data [4–6]. The first concept is to do some preliminary analysis just in the data collecting devices. In this way these devices send the partially analyzed data to management stations, instead of the entire set of monitoring information. The increased intelligence refers also to replacing periodical information sending with reporting performed only upon predefined events occurrence [2].

Along with the mentioned device intelligence increase, changes in the way of collected data reporting are also being proposed. An example of such a solution is the method of Intrinsic monitoring [7–9]. Its major feature is that it can significantly reduce the generated network traffic [7]. The principle of this method involves the use of an additional IPv6 header and the transfer functionality to decide about sending measurement data to network devices. Their task is to make autonomous decisions in order to send the measurement data, e.g. through the use of existing traffic (packets) in the network.

Section 2 describes the character of network monitoring, which is carried out within the framework of netBaltic project [10,11]. The essence of this network is that wireless links with a relatively low bit rates and high error rate are the only available. Section 3 contains a description of analyzed solutions about how to transport the collected monitoring data. In this section also the EHBMPvU and EHBMPvF methods are presented. Section 4 is devoted to the comparison of the developed methods and the classic solution to transport monitoring data. Finally, Sect. 5 presents the results of the analysis along with the indicators when and in what conditions should the particular method of reporting be used.

2 The Problem of Monitoring Specific Mesh Network Topology

The netBaltic Project can be characterized as a heterogeneous, wireless, self-organizing network with multi-hop transmission. In general there are wireless links with relatively low bitrates and high error rates mainly due to wavy water surface interacting with electromagnetic waves. Therefore, an important element is to ensure the least possible bandwidth.

Reporting of measurements, that is, transporting of the collected data via the network to the management stations, requires an additional traffic, which affects the behavior of the network. In case of netBaltic system, it is important that the monitoring system should provide supervision of all the network elements. This means that the entire network should be monitored. Another important thing is that the monitoring solution implementation should pose the least possible overhead. Such assumptions stem from the nature of the network.

The aim of the netBaltic Project is to develop innovative mechanisms for self-organizing heterogeneous wireless networks, allowing possibly fast data transfer between vessels, ships and centers of storage and data processing, as well as the public Internet access. Wireless data transmission at sea will improve the safety of navigation through the realization of e-navigation services [10]. This project

is being implemented by a consortium whose leader is Gdansk University of Technology. The netBaltic system is intended to provide constant connectivity with ships and vessels from the Internet. In case this is not possible, to provide certain services operating in delay tolerant mode (e.g. maps updates, weather forecast delivery, e-mail). It is important that the construction of the network structure will be based on IPv6 protocol.

Due to the nature of the network and the characteristics of sea physics it is essential to employ appropriate mechanisms for routing between nodes. The priority is to minimize the consumption of bandwidth resources by both the signaling (routing) and user data selection strategy, to avoid unnecessary transmission. Hence, each netBaltic system node functions as a specialized router.

The routing solution proposed by the netBaltic Project consortium, is to use two complementary routing protocols, i.e. proactive: TBRP (Tree-Based Routing Protocol) [12] and reactive: RM-AODV (Radio Metric Ad-hoc On Demand Distance Vector) [12]. This combination of two complementary types of routing can be called a hybrid routing. Such combination, known from the IEEE 802.11s standard [13], is called Hybrid Wireless Mesh Protocol. TBRP routing protocol employs a tree structure and does not provide full knowledge about the network topology to each node. When TBRP routing does not guarantee knowledge of the route between nodes, it is necessary to use the RM-AODV method, which is a reactive routing protocol, used on demand.

3 Network Monitoring System Proposal

The bandwidth of NetBaltic system is extremely valuable and should be used reasonably. For this purpose, this paper contains two different proprietary solutions, both based on the mechanism of the additional IPv6 headers.

The first proposal for the monitoring system is called EHBMPvF (Extension Header Based Monitoring Protocol version Forced), while the second proposal is EHBMPvU (Extension Header Based Monitoring Protocol version Unforced). Both solutions of transport monitoring data guarantee full network coverage [14].

3.1 EHBMPvF Method

EHBMPvF monitoring works by sending special monitoring packets, which aim is to visit specified list of nodes in a predefined sequence. The last address in the list is the node from which the monitoring packet was sent. Single so-prepared package allows to collect data from a certain number of nodes, depending on the data link layer payload size and the size of monitoring data acquired from each node.

Monitoring messages can be sent at a specified time interval or on demand. Monitoring is initiated from the network node that is monitoring center. The principle of monitoring EHBMPvF is based on the additional routing header (type 0) available in IPv6. This type of additional header, employing IPv6 addresses, allows to define which network nodes packet has to traverse along its route.

The mechanism of formatting lists of nodes' addresses for packet monitoring works in a similar way to the BFS (Breadth-First Search) tree (graph) – first the nodes closest to the monitoring center are visited. As the TBRP proactive routing protocol produces routing tables with just the number of hops to reach each node (network prefix in general), this information is used to prepare the list of nodes to visit in the specified order – from the closest to the most distant ones.

All of these routers' addresses in the header must be visited in the specified order, but other nodes may be visited on the way. To specify the list of nodes, through which a monitoring packet is expected to traverse, it is necessary to use information kept by the node that is the root of the tree, as it has information about how to get to all the nodes assigned to it.

When a node receives such packet, it takes action aimed at checking whether the node is the recipient of this message, i.e. the IPv6 datagram's destination address.

If it is not, the package is sent further, according to the route determined by the routing mechanism. As there is the possibility that the node would not know the route to the next node specified, the reactive routing protocol RM-AODV is used to find the route.

If the receiving node belongs to the list of specified addresses, it adds own monitoring data to the received packet. The data is structured as in Fig. 1 and belongs to the datagram payload.

Fig. 1. The data structure placed in a package by the monitored node. In parentheses is the length of the data fields in bits.

The fields' meaning is the following:

- TimeStamp – this field contains the time of transmitted data. Using a 32-bit the information about the year (12 bits), month (4 bits), day (5 bits), hour (5 bits) and minutes (6 bits) can be stored;
- Fragment Length – length in bytes of Data field (16 bits);
- Fragment ID – this is a field informing that the Data field carries only part of the whole monitoring information. The value identifies the number of the fragment. The value of 0 indicates that the data placed in the structure is not fragmented (16 bits)
- Data – contains the collected data from a single node.

Along path traversal each such structure is added, successively one after another, reflecting the order of nodes along the path specified in the header.

Changing Network Conditions Mitigation. The EHBMPvF monitoring uses the information contained in the TBRP root's routing table of the tree, which is updated in discrete time periods. Therefore there is possible situation in which the node being on the list to visit, either loses the possibility to communicate, or fails completely, leading to the loss of even indirect communication with off-shore infrastructure. This means that both the TBRP and RM AODV protocols won't be able to find the route. In such situation a modification of the routing header should be made, by removing the node from the list and sending the packet to the next node in the list.

Another undesirable situation that may occur, is when one of the nodes loses packet with monitoring data as a result of an accident. To overcome this, the TCP protocol is used for sending the monitoring packets for reliability. In case of damaged link used in the created TBRP tree, besides using tree reconstruction mechanisms, the other option is to perform RM AODV routing.

All these mechanisms are employed to ensure that the EHBMPvF monitoring solution is resistant to changes in the network, i.e. upon a node failure, or topology change, the monitoring will continue reacting to current conditions.

Network Coverage in EHBMPvF. Each special packet sent by the monitoring EHBMPvF contains the list of nodes to visit and to collect data from. The network transfers this packet until each node listed adds its monitoring data, and then the packet sent back to the monitoring center. The TCP protocol is used for each step, which ensures that the transfer of packet is successful. In this way the monitoring center obtains data from the specified part of the network – the number of nodes is equal to the size of the address list.

Therefore, using the rule of deduction, it can be concluded that if one packet allows for partial coverage of the network, then sending P packets with unique node addresses on the list, the full network coverage can be achieved. This number is determined by the following formula:

$$P = \left\lceil \frac{x}{\left\lfloor \frac{w}{z} \right\rfloor} \right\rceil, \tag{1}$$

where:

P – the required number of monitoring packets,
x – total number of nodes to be monitored (value representing the size of the routing table TBRP at the root of the tree),
w – the maximum size of payload data in the application layer of a single datagram in bytes,
z – the size of data collected from each node in bytes (the size should be shared by all the nodes).

The P value is calculated based on the current number of available network nodes, taken from the routing table. Hence, the full network monitoring coverage by the EHBMPvF method can be obtained, if and only if the monitoring center sends P special packages.

3.2 EHBMPvU Method

Collecting data from the nodes in the network using the EHBMPvU monitoring method involves the use of existing users' packets as the mean of transport. The packet have to be addressed to the monitoring center's IP address. One packet allows to collect data from a certain number of nodes, depending on the size of the data itself and available application layer payload space.

The rate of sending data depends on the frequency of nodes' applications communication. Monitoring is initiated by each node independently. The principle of operation in monitoring EHBMPvU is based on an additional new header, named Monitoring Header, which aims to transport the data required for monitoring.

Adding data to the package is possible if and only of the three conditions are met:

- packet must have a valid recipient. To reach the target data, it can only be added to a packet, which will meet on its path the monitoring center node;
- the packet have enough space to add an additional header structure either with the data or the data itself, if the header has been added by some previous node along route;
- there must be the need to send monitoring data.

Monitoring Header and method of its placement in the packet is shown in Fig. 2.

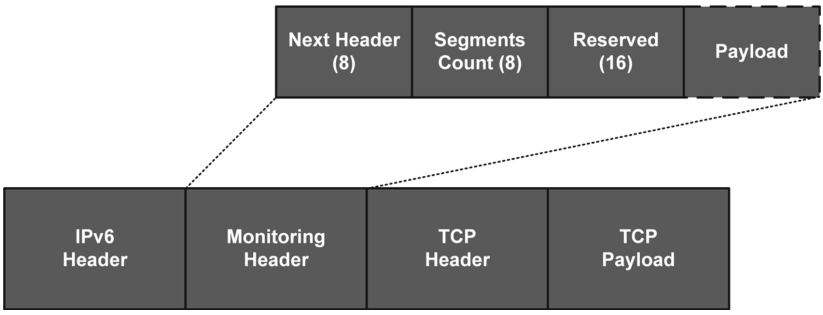


Fig. 2. Additional Monitoring Header and scheme of placing it in the datagram. In parentheses is the lengths of the fields in bits.

The proposed Monitoring Header, which is shown in Fig. 2 has the four following fields:

- Next Header – identifies the next header following just after the Monitoring Header. This field allows for compliance with the mechanism of the additional headers in IPv6 (8 bits);
- Segments Count – determines the number of nodes, which added data to the Monitoring Header. When adding data, increase this value (8 bits);

– Reserved – field that was introduced to take into account possible future modifications to this header. Currently it is being skipped during processing (16 bits);
– Payload – field in which structure are added – contain data added by the node, including monitoring data.

Figure 3 shows the data structure which is placed in the Monitoring Header. Each new structure is included consecutively one by one.

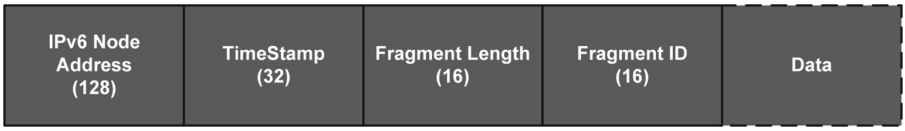


Fig. 3. The structure of the data placed in the Monitoring Header by the node. In parentheses is the length of the data fields in bits.

Meaning of the fields of this structure is similar to the structure shown in Fig. 1 used by the EHBCMPvF method. There is an additional IPv6 Node Address field at the beginning in the length of a single IPv6 address (128 bits), which is used to identify the origin of the data. This field contains the IPv6 address of the node which added the data to the packet.

After adding the collected data to the packet, the node sends it towards the monitoring center. Each packet which has an additional header can carry the monitoring data from at least one node. The size of monitoring data depends on meeting the three previously described conditions.

Changing Network Conditions Mitigation. The EHBMPvU monitoring method is resistant to changes in the network, i.e. when a node failure occurs, or it loses even indirect communication with offshore infrastructure, monitoring will continue without this node. This is due to the fact that a damaged node has no influence on the monitoring process of the other nodes in the network.

The only undesirable situation that may occur is when one of the intermediate nodes on the packet route towards the monitoring center lose packet with monitoring data as a result of an accident. Such situation, however, is addressed by the use of only TCP application protocol for piggybacking purposes. In case of damaged link used in the created TBRP tree, besides using tree reconstruction mechanisms, the other option is to perform RM AODV routing.

Network Coverage in EHBMPvU. Each node in the network operating the EHBMPvU monitoring method is obligated to make an attempt to send monitoring data in preconfigured periods of time. Prior sending the data all the required datagram conditions have to be met, especially the destination address and the available payload space.

The use of just TCP segments ensures that the transfer of packet between the nodes will be reliable. Finally, the packet reaches the target and monitoring center receives monitoring data from at least one node. Using the rule of deduction, it may be concluded that if each node in the network behaves as described above, full network coverage can be achieved. The worst case scenario requires using the number of application packets equal to the number of nodes in the network. The necessary number of packets can be specified as follows:

$$P \leq x \tag{2}$$

where:

P – required number of monitoring packets,
x – the number of nodes in the network.

4 Comparison of the Presented Methods with the Classic Method of Monitoring

In order to determine the efficiency of two proposed methods for the netBaltic system, the simulations was performed by using the devoted simulator [14] and the results were compared to the typical monitoring use case – namely the SNMP protocol. The latter was assumed to work in the send request and wait for response mode of operation. Scalability, security and the overhead were taken under consideration in the evaluation.

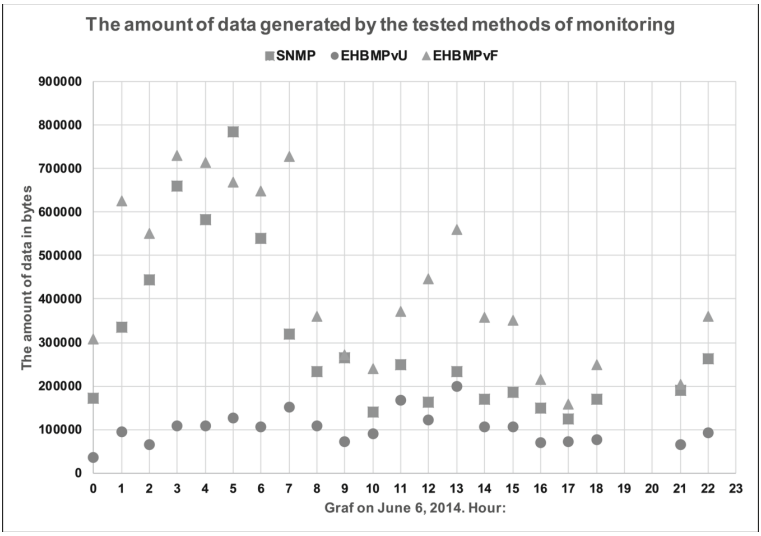


Fig. 4. Comparing the amount of data generated by the methods EHBMPvF, EHBMPvU and SNMP (classic method).

The experiment was performed on the data collected from the AIS system by Baltica ship, dated on 6th June, 2014 [15]. The simulation was performed for snapshots of ships' positions taken in one hour intervals. Summary, which provides information about numbers of nodes and links, as well as the sizes of the root's arrays of the trees (obtained by the TBRP protocol) for particular hours, are shown in Table 1. The data for hours 19, 20 and 23 was unavailable, because the node selected for the root of a tree in TBRP routing, did not contain any record in its routing table. This is due to the fact that the data collected by the AIS system by Baltica ship does not necessarily show all the ships in the Baltic Sea region, because this system has limited range.

For each network snapshot, 15 iterations of the simulation were run during which data transport methods used for monitoring were analyzed. The results, which represent the total amount of generated data on all links of the network are shown in Fig. 4.

Table 1. Hourly summary for number of vessels (nodes in the network) that have been registered by the AIS.

Time (hour)	Number of nodes	Number of links	The size of root's routing table
0	124	3024	99
1	120	2604	116
2	121	3079	112
3	136	3427	134
4	135	3566	134
5	138	3382	136
6	139	3613	130
7	133	2559	126
8	75	685	67
9	60	352	54
10	57	237	47
11	68	339	58
12	64	306	58
13	70	395	64
14	63	281	55
15	59	253	52
16	60	268	45
17	50	200	38
18	54	279	47
21	58	271	47
22	66	371	59

To perform monitoring using EHBMPvF method, the reactive routing RM-AODV was used, due to the fact that the TBRP routing did not provide the knowledge about the whole structure of connections in the network available at each node. Therefore, the results of EHBMPvF should be considered together with the RM AODV overhead.

The evaluation of EHBMPvF monitoring just without taking into account the effect of RM-AODV protocol resulted in generating about 354% more data on average in related to the EHBMPvU method. Comparing it with the classic method gave on average about 61% more data. On the other hand, the classic method generated on average about 233% more data comparing to the EHBM-PvU method. Worth mentioning here is the fact, that typically SNMP monitoring requires several send/receive operations for each OID separately. In our case it was assumed to be a single request and single bulk response in the size of 128 bytes with monitoring data, which was equal for the other methods too.

The results taking into consideration the amount of data generated also by the RM-AODV routing, it can be concluded that the EHBMPvF monitoring required to generate on average about 17,600% of the data compared to the EHBMPvU method and 4290% to the classic one. These are the worst case values, because in typical scenario the network would also employ the RM-AODV for application data routing, so it would reduce the number of on demand routing calls dedicated purely to monitoring requests, thereby reducing the total amount of data generated.

Scalability is a very important feature, which allows to assess whether the future system expansion has a chance to succeed or fail. Analyzing the results obtained in simulation environment, it was agreed that the EHBMPvU scales much better than EHBMPvF monitoring. The classic method for network monitoring, known as send a request and wait for the response, is a popular solution, but for the tested networks, it generated on average about 230% more data compared to the EHBMPvU.

The actual, precise time consumed for collecting data from the whole network (using presented monitoring methods) is impossible to determine, because of multitude of factors that may run in parallel in the real system. However, this time can be estimated.

In the case of EHBMPvF implementation, the duration of the monitoring process consists of the time required for propagation of special data packets over the network and delays related to the need for on demand RM-AODV routing. The more times the RM-AODV on demand routing is called, the more total time is required for EHBMPvF monitoring.

However, in the EHBMPvU method, besides the time associated with the propagation of packets from a node to the monitoring center, significant influence has the time to wait for the right TCP packet. The probability of occurrence an appropriate packet (which can be used to transport the collected measurement data) at the node at time T after the occurrence of the request specifying the need to send the collected data can be represented by the equation:

$$P(A) = P(Y) \int_0^T p(x)dx \int_1^R h(z)dz \tag{3}$$

where:

$P(A)$ – the probability of an event A, the occurrence of the proper packet;
$P(Y)$ – the empirical probability of (derived from statistical surveys) the event y, where y event states that the packet is either addressed to the monitoring center or has it along its route;
$p(x)$ – empirical probability distribution, which specifies the possibility of passing through or sending the packet by the node, where x is the packet arrival time;
T – time in which the proper packet occurs;
$h(z)$ – empirical probability distribution which define the probability of a packet that has a specified size of data field of link layer protocol, where z is the size of the data field;
R – maximum size of the data field, decreased by size of added monitoring data.

The monitoring time in the classic version of the monitoring depends only on the delay associated with the propagation of data packets from the monitoring center to a node, and then back to the monitoring center.

The proposed EHBMPvU monitoring mechanism is not transparent to the devices in the network, as it requires all devices to support the use of the proposed additional IPv6 header mechanism. If all the nodes in the network have to be monitored, this means that all the nodes must be able to recognize and process the additional monitoring header. Otherwise the packet is discarded by the router, because the node will not know what to do with such header. In case of the EHBMPvF mechanism, these issues looks different, because the Routing Header (Type 0) is described in IPv6 [16], but for safety reasons, in the year 2007, the Routing Header (Type 0) was withdrawn and marked by status of disapproval (deprecated) [17]. Therefore, it may not be supported by the routers. This is due to the fact that this header can be used to perform DoS (Denial of Service) attacks. The issue has been solved by introducing Routing Header Type II [18].

In the context of the necessary requirements to perform monitoring, one should keep in mind the resources of devices in the network. The described monitoring solutions require the ability to parse, process and send monitoring data, so routers must have sufficient processing power to perform these tasks, along with the other functions implemented in router. The busiest router is node which is the monitoring center, because it collects data from all nodes in the network. In the EHBMPvF method it is also responsible for sending requests for collecting data. To send such packets, the router must perform additional calculations associated with the optimization of the number of required packets and must allocate a list of addresses to visit for each packet. The EHBMPvF mechanism uses the RM-AODV on demand routing, which generates additional traffic in the network and load to the nodes.

Hardware requirements for the classic method of monitoring are similar to what EHBMPvU requires – the processing of monitoring data.

Security of collected data is also important as they should be kept confidential and integral. To ensure these requirements, the use of IPsec (Internet Protocol Security [19–21]) security suite is recommended in the netBaltic Project.

5 Summary

To summarize the presented characteristics in Table 2, one can come to the conclusion that the best solution is to use EHBMPvU monitoring, which is well scalable, provides full coverage in a short time, has small requirements and is secured by the means for confidentiality and authenticity. The only problem which may occur is the case when the application traffic characteristics would a node require to wait long for the correct data packet. In the extreme case where the traffic would be strictly limited or the total lack of it, the realization of EHBMPvU monitoring would not be possible.

Knowing the characteristics of the traffic the probability of sending monitoring data by each node can be estimated – after an event triggering this process (at certain time or in the response to the identified event in the network).

EHBMPvU method should be used in networks where the characteristics of the existing traffic does not cause unacceptable delays in providing data used for monitoring. Longer delays may cause that the transmitted data can be regarded as outdated.

The classic method of monitoring is the best choice if the network to be monitored has an unacceptable traffic characteristics to use by the EHBMPvU solution.

Table 2. Comparison of the characteristics of the analyzed monitoring methods.

	EHBMPvF	EHBMPvU	Classic monitoring
Scalability rating (10^n nodes)	n = 1	n = 1, 2, 3, 4, 5	n = 1, 2, 3, 4
Full network coverage	Yes	Yes	Yes
Real time to complete network coverage	From a few milliseconds to several minutes, hours – depending on the characteristics of the existing network traffic	From a few milliseconds to several minutes, hours – depending on the characteristics of the existing network traffic	From a few to hundreds of milliseconds, depending on the size of the network
Hardware requirements (the number of operations performed)	Support monitoring traffic and RM-AODV	Support monitoring traffic	Support monitoring traffic
The security of transmitted data	Yes	Yes	Yes

Alternatively, you can use a hybrid solution using the classic method of monitoring wherever the characteristics of traffic does not allow for acceptable time to provide data to the monitoring center and EHBMPvU where the time is acceptable.

The choice of monitoring method for netBaltic system requires testing the characteristics of mobility. Then and only then it will be possible to unambiguously identify the best monitoring method for this type of network. However, yet at this early stage of testing, the EHBMPvF method can be rejected due to very large overhead related to monitoring.

Acknowledgments. This work has been partially supported by the Applied Research Program under the Grant: ID PBS3/A3/20/2015, founded by the National Centre for Research and Development.

References

1. Silvestri, S., Urgaonkar, R., Zafer, M., Ko, B. J.: An online method for minimizing network monitoring overhead. In: 2015 IEEE 35th International Conference on Distributed Computing Systems (ICDCS), pp. 268–277. IEEE (2015)
2. Lee, S., Levanti, K., Kim, H.S.: Network monitoring: present and future. Comput. Netw. **65**, 84–98 (2014)
3. Mahkonen, H., Manghirmalani, R., Shirazipour, M., Xia, M., Takacs, A.: Elastic network monitoring with virtual probes. In: 2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN), pp. 1–3. IEEE (2015)
4. Prieto, A.G., Stadler, R.: A-GAP: an adaptive protocol for continuous network monitoring with accuracy objectives. IEEE Trans. Netw. Serv. Manage. **4**(1), 2–12 (2007)
5. Dilman, M., Raz, D.: Efficient reactive monitoring. IEEE J. Sel. Areas Commun. **20**(4), 668–676 (2002)
6. Jiao, J., Naqvi, S., Raz, D., Sugla, B.: Toward efficient monitoring. IEEE J. Sel. Areas Commun. **18**(5), 723–732 (2000)
7. Höfig, E., Coşkun, H.: Intrinsic monitoring using behaviour models in IPv6 networks. In: Strassner, J.C., Ghamri-Doudane, Y.M. (eds.) MACE 2009. LNCS, vol. 5844, pp. 86–99. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-05006-0_7
8. Shi, L., Davy, A., Muldowney, D., Davy, S., Höfig, E., Fu, X.: Intrinsic monitoring within an IPv6 network: mapping node information to network paths. In: 2010 International Conference on Network and Service Management (CNSM), pp. 370–373. IEEE (2010)
9. Shi, L., Davy, A.: Intrinsic monitoring within an ipv6 network: relating traffic flows to network paths. In: 2010 IEEE International Conference on Communications (ICC), pp. 1–6. IEEE (2010)
10. Hoeft, M., Gierłowski, K., Nowicki, K., Rak, J., Woźniak, J.: netBaltic: enabling non-satellite wireless communications over the baltic sea. IEEE Commun. Mag. **5** (2016). http://gcn.comsoc.org/netbaltic-enabling-non-satellite-wireless-communications-over-baltic-sea
11. Woźniak, J., Hoeft, M.: Aim and main research tasks of the netBaltic project (in Polish). Telecommun. Rev. Telecommun. **12**, 1301–1303 (2016)

12. Institute of Electrical and Electronics Engineers: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. 802.11-2012 (2012)
13. Institute of Electrical and Electronics Engineers: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications - Amendment 10: Mesh Networking. 802.11-2011 (2011)
14. Karpowicz, D., Nowicki, K.: Implementation of database API for archival large-scale AIS data retrieval for netBaltic Project simulation purposes (in Polish). Scientific report, Gdańsk (2016)
15. Lewczuk, M., Hoeft, M., Cichocki, P., Woźniak, J., Nowicki, K.: Systems of AIS data acquisition, processing and visualization for the netBaltic project (in Polish). Telecommun. Rev. Telecommun. News **12**, 1326–1329 (2016)
16. Deering, S., Hinden, R.: RFC 2460, Internet Protocol, Version 6 (IPv6) Specification. Internet Engineering Task Force (1998)
17. Neville-Neil, G., Savola, P., Abley, J.: RFC 5095, Deprecation of Type 0 Routing Headers in IPv6 (2007)
18. Johnson, D.B., Arkko, J., Perkins, C.E.: RFC 6275, Mobility Support in IPv6 (2015)
19. Seo, K., Kent, S.T.: RFC 4301, Security Architecture for the Internet Protocol (2005)
20. Kent, S.T.: RFC 4303, IP Encapsulating Security Payload (ESP) (2005)
21. Gierszewski, T.: Transport security mechanisms for netBaltic system (in Polish). Telecommun. Rev. Telecommun. News **8–9**, 813–816 (2017)

A Novel Auction Based Scheduling Algorithm in Industrial Internet of Things Networks

Mike Ojo[1], Stefano Giordano[1], Davide Adami[1,2], and Michele Pagano[1(✉)]

[1] Department of Information Engineering, University of Pisa, Via Caruso 16, 56122 Pisa, Italy
mike.ojo@ing.unipi.it, {s.giordano,m.pagano}@iet.unipi.it
[2] CNIT Research Unit, Galleria Gerace 18, 56100 Pisa, Italy
davide.adami@cnit.it

Abstract. Enabling low data losses and ensuring reliability are essential requirements to guarantee Quality of Service in Industrial Internet of Things (IIoT) applications. This can be achieved by the adoption of various architectures and standards, the most promising of which is IEEE 802.15.4 Time Slotted Channel Hopping mode. However, there are still several open issues, such as providing effective scheduling scheme in the standard. In this paper, we propose a novel auction based scheduling mechanism that uses a first-price sealed bids auction to solve the throughput maximizing scheduling problem. We considered a centralized IIoT network where the gateway makes frequency allocations and time slot assignment. Simulation results show that the proposed algorithm yields a very close throughput performance to the optimal one obtained through CPLEX with a much lower complexity.

Keywords: Industrial Internet of Things · TSCH · Auctions
IEEE 802.15.4-2015 · Centralized scheduling · Resource allocation

1 Introduction

The notion of the Industrial Internet of Things (IIoT) has moved beyond its hype of becoming the next wave of innovation, and today we are seeing a rapid increase in deployment of IIoT in many application areas, such as smart farming, smart grids, building automation. IIoT relies on low-cost, low power wireless sensor and actuator devices with constrained computational capabilities that are connected to the internet through wireless communication technologies [1,2]. The deployment of such devices has led to the convergence of operational technologies (i.e., industrial networks) and information technologies (i.e., internet). The principle of IIoT has also fostered the concept of Industry 4.0, where it is aimed to provide automation and information domain for services and people [3].

The increasing usage of wireless technologies in industrial environments has become appealing to many types of applications. This has introduced flexibility and a cost-effective solution in the network approach, but has also added

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 103–114, 2018.
https://doi.org/10.1007/978-3-319-92459-5_9

numerous challenges, such as means to achieve wired-like reliability, security, performance guarantees, low power consumption, just to mention a few. The challenges of using wireless technologies in performing different industrial tasks is being addressed by standards such as IEEE 802.15.4.

IEEE 802.15.4 is one of the leading standard in IIoT and it defines the physical (PHY) and the Medium Access Control (MAC) layers for low power, low rate wireless personal area networks. The third revision of the standard called IEEE 802.15.4-2015 [4] was released to address the limitations of its previous releases. IEEE 802.15.4-2015 presents Time Slotted Channel Hopping (TSCH), which provides high reliability by mitigating the effects of interference and multipath fading through channel hopping, and ensures low power consumption through time synchronization to various industrial applications running in harsh environments. IEEE 802.15.4-2015 TSCH mechanism has been adopted into the industrial wireless standards such as ISA.100.11a [5] and WirelessHART [6]. Figure 1 shows the architecture of the proposed IEEE/IETF standardized IIoT.

The standard defines the mechanism of how TSCH executes a schedule, however, it does not mandate a specific scheduling mechanism, which is open to different implementations from the industry. In this paper, we propose an auction theory based scheduling method to address the throughput optimal scheduling problem formulated in [7]. The scheduling model in [7] was slightly modified by introducing multiple antennas, improving communication reliability. The proposed scheduling model is aimed at accomplishing many tasks at the same time such as frequency, time slot, data rate allocation in a heterogeneous multi-user and multi-channel environment. The scheduler, which is executed by the gateway, maximizes the total throughput of the nodes in the service area while ensuring that each node is assigned at least one time slot in any scheduling period, with no collisions occurring among the nodes.

The remainder of this paper is organized as follows: Sect. 2 discusses the basic concept about TSCH Mechanism and the related work. Section 3 introduces the system model and the problem formulation. Then, Sect. 4 proposes a novel scheduling approach based on auctioning. We discusses about the simulation results in Sect. 5 and finally in Sect. 6, we conclude the paper with some final remarks.

2 Background and Related Works

2.1 IEEE 802.15.4-2015 TSCH Concept

All nodes are synchronized in TSCH networks and communication occurs at well-defined times within a time slot. A typical time slot is sufficient to transmit a single frame and receive an acknowledgement. A slot frame is a group of time slots that repeats over time and can also be described as time-frequency offset channel distribution unit (CDU), where each cell represents a fixed time slot in a specific channel offset. Each node in the network only cares about the cell it participates in.

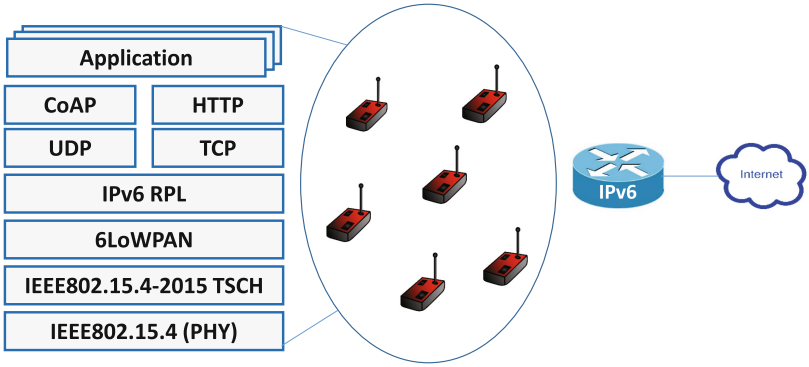


Fig. 1. IEEE/IETF standardised IIoT architecture

TSCH communication is orchestrated by a scheduler, where a schedule is defined by the time slot and channel on which a node should communicate with its neighbours. TSCH schedule can be represented by a 2-D slot frame-channel offset matrix as shown in Fig. 2. The two nodes at either end of the link communicate periodically once in every slot frame. To improve communication reliability, TSCH proposes to implement channel hopping, with frequency diversity to mitigate the effect of narrow-band interference and multipath fading. The channel offset is translated into a frequency f (i.e., a actual channel) using the following frequency translation function

$$f = F_{MAP}(ASN + \text{channeloffset}) mod N_{ch} \tag{1}$$

where ASN (Absolute Slot Number) counts the number of time slots since the beginning of the network operation, F_{MAP} is the mapping function to find the frequency from a channel lookup table and N_{ch} indicates the number of available physical channels (e.g., 16 when using IEEE 802.15-4 compliant radios at 2.4 GHz).

2.2 Related Works

Several schedule-based MACs have been proposed in the wireless networking literature, but they can not be applied to the TSCH MAC. Nonetheless, TSCH paradigm brings this topic into the focus of research again due to the following reasons: (a) IEEE 802.15.4-2015 standard defines the mechanism for a TSCH node to communicate, but the standard does not specify how to build an optimized schedule and to construct a schedule is policy specific; and (b) TSCH brings new opportunities and challenges because of its time synchronization multiplexed in frequency to improve reliability.

Only few works have focused on scheduling in TSCH networks. The pioneering work [8] proposed by Palattella et al. focused on a centralized approach based on matching and colouring of graph theory to plan the distribution of time slot

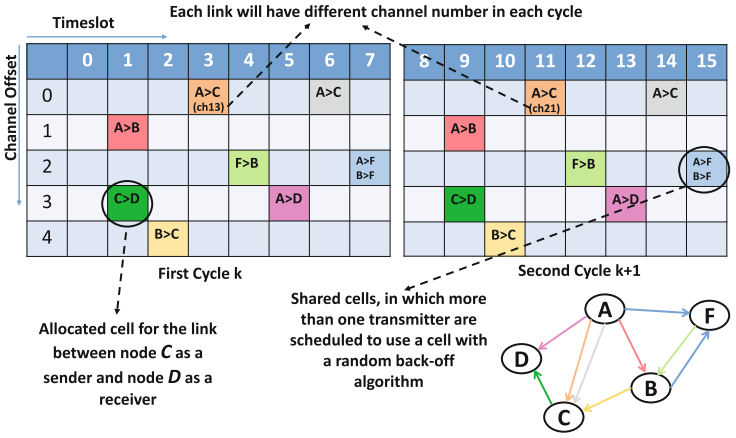


Fig. 2. Time Slotted Channel Hopping (TSCH) slot-channel matrix with a simple topology

and channel offset. A distributed scheduling approach [9] was also proposed to construct optimum multi-hop schedules based on neighbour-to-neighbour signalling. Another non-graph approach for scheduling is Orchestra [10], a solution for autonomous scheduling of TSCH in IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL). Orchestra runs without any central entity, nor any negotiation, but allocates the slots in such a way that it can be automatically installed or removed as the RPL topology evolves. In [11], the authors proposed a scheduling scheme to maximize the energy efficiency, while [12] addresses latency issues.

In this paper, we focus on a centralized scheduling method based on auctioning to maximize the total throughput in an interference-aware system, since a centralized approach have been proven to be more effective in practice [13]. The major novelty of our work is that we provide a much more general and complete scheduling model that achieves frequency, time slot, data rate allocation in a heterogeneous multi-user and multi-channel environment. Our auction procedure uses a first-price sealed-bid mechanism [14] to address the throughput scheduling problem in IEEE 802.15.4-2015 TSCH networks. In first-price sealed-bid auctions, all bidders simultaneously submit sealed bids (hidden from other bidders) to the auctioneer. The bidder with the highest bid wins the auction.

Auction based mechanism have been applied to other networks such as cognitive radio networks [15] cellular networks device to device communication systems [16] peer to peer networks [17], but to the best of our knowledge, this is the first work that addresses scheduling in IEEE 802.15.4-2015 TSCH networks using auction based mechanism.

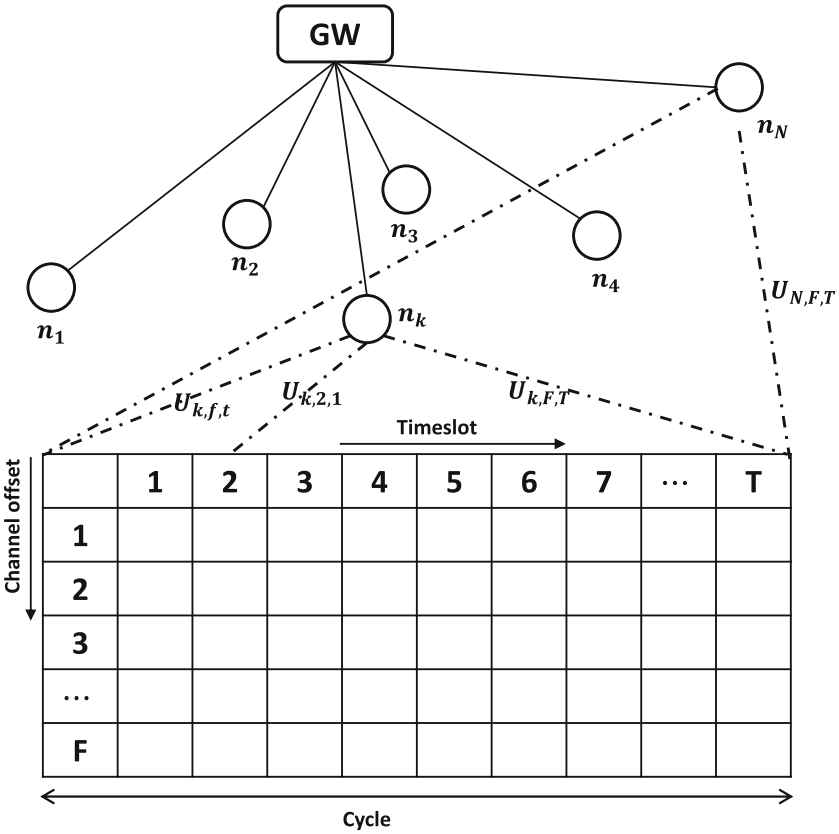


Fig. 3. Time Slotted Channel Hopping (TSCH) slot-channel matrix where each n_k maintains a link with the gateway (GW) for both frequency f and time slot t, where $k \in \{1, ...N\}$; $f \in \{1, ..., F\}$, $t \in \{1, ..., T\}$

3 System Model and Problem Formulation

We consider a time-slotted IEEE 802.15.4-2015 network, where the nodes are managed by the gateway in a centralized manner as shown in Fig. 3. The network consists of static nodes, and their locations are assumed to be known. Each node is equipped with a radio with a communication range R_f and the radio is assumed to transmit at a fixed transmission power. In addition, different channels can support different link capacities and transmission ranges. We are given a set of nodes n_k where $k \in \{1, ..., N\}$, characterized by its frequency $f \in \{1, ..., F\}$ and its time slot $t \in \{1, ..., T\}$. A communication graph $G(V, E)$ is used to model the network, where every node $k \in V$ corresponds to a device in the network, and there is a link $e = (u, v) \in E$ if there exists a common channel available for between u and v. Transmission is successful if the distance between them $\|u - v\| \leq R_f$ condition is satisfied.

At the beginning of the slot frame, each node n_k calculates the transmission capacity of the channel for each available frequency, and transmits this information to the gateway in addition to the number of packets in its buffer (Q_k). In this way, the gateway constructs a matrix $U = [U_{k,f,t}]$, where $U_{k,f,t}$ is the number of packets that can be transmitted by node k using frequency f. Upon collection of all state information such as topology information, traffic generated by each node etc., the scheduler decides on the frequency assignment with the goal of maximizing the throughput.

Let $C_{k,f,t}$ denote the Shannon's capacity of the link between n_k and the gateway GW at frequency f. $C_{k,f,t}$ is a function of the signal-to-noise ratio $SNR_{k,f,t}$ and represents the theoretical upper bound. In the calculation of $C_{k,f,t}$, only the background noise is considered as we focus on an orthogonal frequency assignment. The number of packets that can be sent during a slot frame duration, $M_{k,f,t}$, equals $C_{k,f,t}T$ where T is the slot frame duration. Accordingly, $U_{k,f,t}$ becomes $min(M_{k,f,t}, Q_k)$.

Considering the binary variable $X_{k,f,t}$ defined as:

$$X_{k,f,t} = \begin{cases} 1 \text{ if node } n_k \text{ transmits using frequency } f \text{ in time slot } t; \\ \\ 0 \quad \text{otherwise;} \end{cases}$$

$\boldsymbol{x} = [X_{k,f,t}, k \in \{1, ...N\}; f \in \{1, ...F\}; t \in \{1, ...T\}]$ is the scheduling decision vector with elements $X_{k,f,t}$. Note that $X_{k,f,t}$ is a function of the information available to n_k. We calculate the total throughput in TSCH network as

$$\mathrm{C} = \sum_{k=1}^{N} \sum_{f=1}^{F} \sum_{t=1}^{T} U_{k,f,t} X_{k,f,t} \tag{2}$$

Interference and spectrum availabilities are constrained in the derivation of the formula for $U_{k,f,t}$. For instance, if a node k is very close to a node i that uses frequency f in time slot t, then node k cannot use frequency f in the same time slot, i.e., $U_{k,f,t} = 0$. $U_{k,f,t}$ value quantifies the slot availabilities (i.e., the higher the $U_{k,f,t}$ value is, the higher the data rate node k can have if it uses frequency f in time slot t). $U_{k,f,t}$ values are input variables for the optimization problem in this paper. In the following, $U_{k,f}$ is used instead of $U_{k,f,t}$ because we assume that all parameters are to remain constant for each time-slot t of the slot frame period. The slot frame period is a time duration in which the network conditions remain fairly stable. For instance, in voice applications, the value of T tends to be fairly large.

The binary integer linear programming is as follows:

$$\sum_{k=1}^{N} \sum_{f=1}^{F} \sum_{t=1}^{T} U_{k,f} X_{k,f,t} \tag{3}$$

s.t

$$\sum_{f=1}^{F} \sum_{t=1}^{T} X_{k,f,t} \geq 1; \quad \forall k \in \{1, ..., N\} \tag{4}$$

$$X_{k,f,t} + X_{k',f,t} \leq 1; \quad \forall k, k' \in N, k \neq k', \forall f, \forall t \tag{5}$$

$$\sum_{f} X_{k,f,t} \leq a_k \quad \forall k; \forall t \tag{6}$$

$$X_{k,f,t} \in \{0,1\}; \quad \forall k \in N; \forall f \in F; \forall t \in T \tag{7}$$

In the problem formulation, the objective function in (3) maximizes the total throughput of all the nodes in TSCH network governed by the gateway. The constraint in (4) ensures that each node is assigned at least one time slot and hence provides temporal notion of fairness. The constraint in (5) is used to avoid collision, by guaranteeing that at most one user can transmit in a certain slot and frequency offset. Constraint in (6) indicates that node k cannot transmit at the same time using more frequencies than the number of its transceivers (antennas) a_k because each transceiver can tune to at most one frequency at a time, and finally (7) indicates a binary decision variable.

We assume in the simulations part of this work that the buffers of the nodes are continuously backlogged; i.e., there are always enough packets to be transmitted with the data rate determined by the scheduling algorithm. This is necessary in order to effectively evaluate the scheduling process performance by avoiding the possible influence of the traffic arrival process. The above formulation does not assert a minimum throughput guarantee for a node's transmission, however it can be extended by simply adding the following constraint for each node and frequency pair: $U_{min} X_{k,f,t} \leq U_{k,f}$, where U_{min} is the minimum throughput.

4 Auction Based Mechanism Scheduler

In this section, we propose an auction based scheduling algorithm to address the optimization problem formulated in (3)–(7). Our motivation for using a first-price sealed-bid auction in designing suboptimal scheduler for the throughput maximizing scheduler is manifold. First, $U_{k,f}$ values of any node k are independent from other nodes' values, and each node knows only its own $U_{k,f}$, namely its bid value. Since the bid values do not affect each other, the auctions are held in a sealed bid fashion. Secondly, in order to maximize the network throughput, first price auctions are used in our algorithm and an auctioned frequency time resource pair ($FTRP$) is assigned to a node whose bid is greater than all other bids. The result of an auction related to an $FTRP$ does not impact another auction which is held simultaneously.

In our TSCH model, we assume that all nodes managed by the same gateway have the same number of transceivers. The auctioning procedure requires three main identifiers which are (1) The auctioned resources (r) (2) The bidders (3) The auctioneer. We indicate the auctioned resources as $FTRP$; the bidders as the nodes in the network; and lastly the auctioneer as the gateway (where the scheduler resides).

If we ignore constraint (4), (6) and (7), the optimal solution is achieved when a $FTRP$ is assigned to the node k that has the maximum $U_{k,f}$ value for the frequency f (of this resource r). The aim is to assign at least one $FTRP$ to each

node, and to avoid any starving node at the end of the algorithm. A starving node is indicated as the node that has not been assigned any time slot during any stage of the algorithm. Our proposed scheduling algorithm is explained through steps 1 to 6 below.

STEP 1: For each frequency f, find the node who transmits the maximum number of packets using that frequency. Assign the frequency to that node for all time slots of the slot frame period. In other words, assign f to node k where $k = argmax_k U_{k,f}$. We denote w_k the number of frequencies assigned to node k at the end of step 1.

STEP 2: If every node is assigned at least one $FTRP$ and $w_k \leq a_k, \forall k$, end. Otherwise, each node that is assigned more than one frequency sorts its frequencies according to their $U_{k,f}$ values. If any node k has w_k greater than a_k, go to step 4 and 5, else go to step 6.

STEP 3: Do we have a starving node? if yes, go to step 4, otherwise go to step 5.

STEP 4: Any node k whose $w_k \geq a_k$ auctions all time slots of $w_k - a_k$ of its frequencies which have the smallest $U_{k,f}$ values. The $FTRPs$ are auctioned simultaneously. The starving node bids to the $FTRP$, whose corresponding $U_{k,f}$ value is the greatest one. When the starving node gets a $FTRP$, it is removed from the auction procedure. The auctions continue until all the starving node gets a $FTRP$ or when no $FTRP$ remains. At the end of the auctioning procedure, if we still have $FTRP$ that are not assigned to any node, they are auctioned out to the nodes that have available transceivers. Otherwise, if there still exists a starving node and all resources are assigned, then go to step 6.

STEP 5: Any node k whose $w_k \geq a_k$ takes a_k of its frequencies with the largest $U_{k,f}$ values. The $FTRPs$ which belong to the remaining $w_k - a_k$ frequencies are assigned greedily to the nodes who have available transceivers.

STEP 6: Any node k that is assigned more than one frequency auctions $w_k - 1$ number of its frequencies which have the smallest $U_{k,f}$ values until either no starving node remains or the auctioned $FTRP$ run out. When all nodes have at most a single frequency, if any starving node exists they auction out their remaining frequencies.

5 Performance Evaluation

A simulation based study was carried out in order to evaluate the performance of the proposed algorithm. We consider a TSCH network consisting of sensor nodes randomly placed over a square area of 100 m × 100 m and a gateway located in the center. The x and y coordinates of each node follow a uniform distribution. Every node is equipped with a radio that has a transmission range of 30 m. $U_{k,f}$ values are different owing to the changing network conditions and are obtained for 3000 slot frame periods in each set of simulations where the average is considered. Each slot frame period consists of 30 time slot, where each time slot has a

duration of $t = 10$ ms. We then solve for the throughput maximizing scheduling problem in (3)–(7). using ILOG-CPLEX [18]. We compare the proposed auction heuristic scheduler with the optimal result obtained through ILOG-CPLEX.

Figure 4 highlights how the average network throughput is affected by varying the number of nodes and frequencies. F_{opt} indicate the number of frequencies used in the ILOG-CPLEX simulations, whereas F_{auc} indicates the number of frequencies used in our proposed auction based heuristic algorithm. It can be seen that the performance of our heuristic is very close to the optimum value in all cases. Moreover, it can also be seen that the average network throughput is almost invariant for varying the number of nodes when F is small (i.e. $F = 3$) in both cases. This is because the number of frequencies available for the nodes is small, which makes no much difference to have increasing number of nodes in the system because almost all of the resources are already occupied by all nodes even when the number of nodes is small. As the number of frequencies increases, the average network throughput grows with the number of nodes. This behaviour does not change up to a threshold where network throughput saturates.

We also point out in Fig. 5 how the number of frequencies and antennas affects the performance of the throughput maximization scheduler as shown. We can notice that increasing the number of antennas of each node in the network only makes sense when there is a certain number of frequencies in the system. For instance, having only one antenna i.e., $a = 1$ has the performance as having multiple antennas i.e., $a > 1$ when the number of frequencies is small (when $F < 8$). On the other hand, there is a significant difference in the throughput when we increase the number of antennas from 1 to 2 if the frequencies are between 8 and 16. The reason for this behaviour is highlighted by constraint (4), where each node is assigned at least one $FTRP$. In order to comply with this constraint, the scheduler which resides at the gateway tends to assign some frequency to the first antenna of each node, and then continue to assign frequencies to other antennas. For instance, in Fig. 5, we assume that $N = 8$, and until the point where $N = F$, the scheduler assigns the frequencies to the first antennas of each node. Increasing the number of antennas has similar effect on the total throughput as increasing the number of nodes. Moreover, between $F = N$ and $F = 2N$, the scheduler assigns the frequencies to the second antenna of each node.

5.1 Computational Complexity

In this section, we compute the computational complexity of our algorithm. At the end of step 1, assigning a $FTRP$ to each node has a complexity of $\mathcal{O}(NF)$ because for each frequency, we determine the maximum number of packets $U_{k,f}$. This is the best case scenario, when every node is assigned at least one frequency. In the worst case scenario, when all the frequencies are assigned to only one node which in practice is very unlikely. We need to take into account the additional complexity of steps 2–6. In step 2, the frequencies are sorted out and the complexity is $\mathcal{O}(F \log F)$. Since we have $N - 1$ starving nodes to bid for the frequencies, it will require $\mathcal{O}((N-1)\log(N-1))$ time to sort out their bids

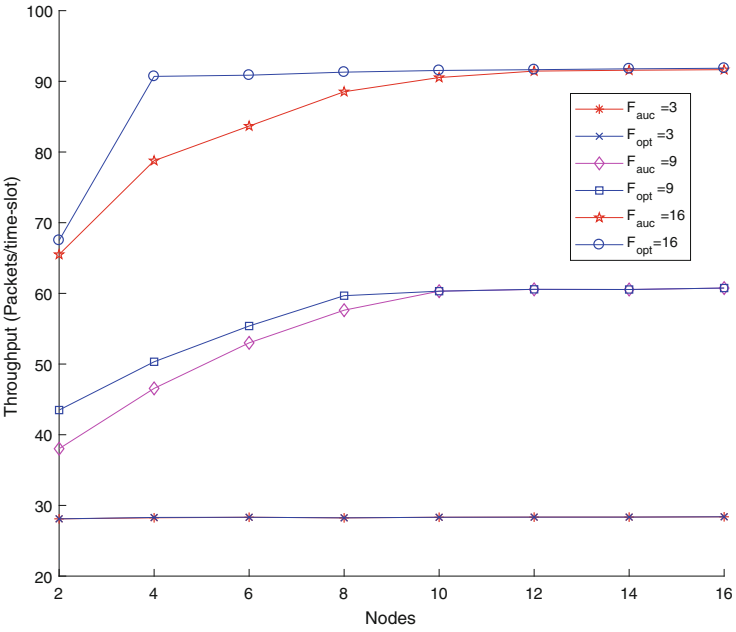


Fig. 4. Comparison of the proposed scheduling algorithm with the optimum results obtained through CPLEX simulations

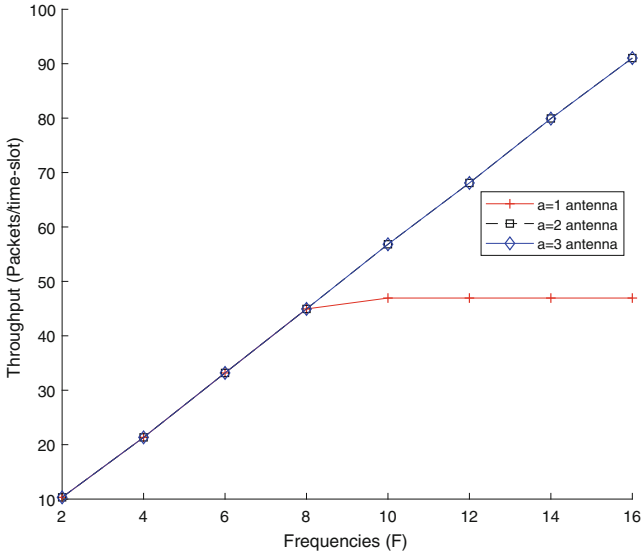


Fig. 5. Average total network throughput for the throughput maximization scheduling problem by varying the number of antennas and frequencies

according to $N-1$ starving node. Therefore the total computational complexity of the algorithm is $\mathcal{O}(NF) + \mathcal{O}(F \log F) + (N-1) \times \mathcal{O}((N-1)\log(N-1))$, that simplifies it to $\mathcal{O}((NF) + N^2 logN)$.

6 Conclusions

In this paper, we formulated the throughput maximization scheduling problem for IIoT-TSCH based networks in a centralized way. In more detail, we proposed a novel heuristic scheduling algorithm based on first-price sealed bid auction mechanism to solve the problem. The performance of our sub-optimal scheduler is close to the optimal one obtained through ILOG-CPLEX. We observed that having many antennas, (i.e., $a > 2$) does not increase the average throughput. Moreover, the computational complexity for the best case scenario is $\mathcal{O}(NF)$ while for the worst case scenario is $\mathcal{O}((NF) + N^2 logN)$.

In our future work, we plan to design approximation algorithms, which have theoretically provide performance guarantee for the throughput maximization scheduling problem.

References

1. Willig, A.: Recent and emerging topics in wireless industrial communications: a selection. IEEE Trans. Ind. Inf. **4**(2), 102–124 (2008). https://doi.org/10.1109/TII.2008.923194
2. Li, X., Li, D., Wan, J., Vasilakos, A.V., Lai, C.-F., Wang, S.: A review of industrial wireless networks in the context of industry 4.0. Wirel. Netw. **23**(1), 23–41 (2017). https://doi.org/10.1007/s11276-015-1133-7
3. Kagermann, H., Helbig, J., Hellinger, A., Wahlster, W.: Recommendations for implementing the strategic initiative INDUSTRIE 4.0: Securing the future of German manufacturing industry; final report of the Industrie 4.0 Working Group. Forschungsunion (2013)
4. IEEE Standard for Low- Rate Wireless Networks: IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011), April 2016
5. ISA-100.11a-2011: Wireless Systems for Industrial Automation: Process Control and Related Applications. ISA.100.11a-2011 (2011)
6. Chen, D., Nixon, M., Han, S., Mok, A.K., Zhu, X.: Wirelesshart and IEEE 802.15.4e. In: 2014 IEEE International Conference on Industrial Technology (ICIT), pp. 760–765. IEEE (2014). https://doi.org/10.1109/ICIT.2014.6895027
7. Ojo, M., Giordano, S.: An efficient centralized scheduling algorithm in IEEE 802.15.4e TSCH networks. In: 2016 IEEE Conference on Standards for Communications and Networking (CSCN), pp. 1–6. IEEE (2016). https://doi.org/10.1109/CSCN.2016.7785164
8. Palattella, M.R., Accettura, N., Dohler, M., Grieco, L.A., Boggia, G.: Traffic aware scheduling algorithm for reliable low- power multi-hop IEEE 802.15.4e networks. In: 2012 IEEE 23rd International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), pp. 327–332. IEEE (2012). https://doi.org/10.1109/PIMRC.2012.6362805

9. Accettura, N., Palattella, M.R., Boggia, G., Grieco, L.A., Dohler, M.: Decentralized traffic aware scheduling for multi- hop low power Lossy networks in the Internet of Things. In: 2013 IEEE 14th International Symposium and Workshops World of Wireless, Mobile and Multimedia Networks (WoWMoM), pp. 1–6. IEEE (2013). https://doi.org/10.1109/WoWMoM.2013.6583485
10. Duquennoy, S., Al Nahas, B., Landsiedel, O., Watteyne, T.: Orchestra: robust mesh networks through autonomously scheduled TSCH. In: Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems, pp. 337–350. ACM (2015). https://doi.org/10.1145/2809695.2809714
11. Ojo, M., Giordano, S., Portaluri, G., Adami, D., Pagano, M.: An energy efficient centralized scheduling scheme in TSCH networks. In: 2017 IEEE International Conference on Communications Workshops (ICC Workshops), pp. 570–575. IEEE (2017). https://doi.org/10.1109/ICCW.2017.7962719
12. Hosni, I., Théoleyre, F., Hamdi, N.: Localized scheduling for end-to-end delay constrained low power Lossy networks with 6TiSCH. In: 2016 IEEE Symposium on Computers and Communication (ISCC), pp. 507–512. IEEE (2016). https://doi.org/10.1109/ISCC.2016.7543789
13. Palattella, M.R., Accettura, N., Grieco, L.A., Boggia, G., Dohler, M., Engel, T.: On optimal scheduling in duty-cycled industrial IOT applications using IEEE802.15.4e TSCH. IEEE Sens. J. **13**(10), 3655–3666 (2013). https://doi.org/10.1109/JSEN.2013.2266417
14. Krishna, V.: Auction Theory. Academic Press (2009)
15. Dong, M., Sun, G., Wang, X., Zhang, Q.: Combinatorial auction with time-frequency flexibility in cognitive radio networks. In: INFOCOM, 2012 Proceedings IEEE, pp. 2282–2290. IEEE (2012). https://doi.org/10.1109/INFCOM.2012.6195615
16. Xu, C., Song, L., Han, Z., Zhao, Q., Wang, X., Cheng, X., Jiao, B.: Efficiency resource allocation for device-to-device underlay communication systems: a reverse iterative combinatorial auction based approach. IEEE J. Sel. Areas Commun. **31**(9), 348–358 (2013). https://doi.org/10.1109/JSAC.2013.SUP.0513031
17. Shneidman, J., Parkes, D.C.: Rationality and self-interest in peer to peer networks. In: Kaashoek, M.F., Stoica, I. (eds.) IPTPS 2003. LNCS, vol. 2735, pp. 139–148. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-45172-3_13
18. IBM ILOG CPLEX: V12. 1: Users manual for CPLEX. Int. Bus. Mach. Corporation **46**(53), 157 (2009)

A New Approach for SDN Performance Enhancement

Madhukrishna Priyadarsini(✉) and Padmalochan Bera

IIT Bhubaneswar, Bhubaneswar, India
{mp18,plb}@iitbbs.ac.in

Abstract. Traditional networks are incapable of reconfiguration to serve heterogeneous traffic and are potentially error-prone. Software Defined Network (SDN) has evolved as an assuring solution to serve a good deal of heterogeneous traffic with varying requirements. The basis of SDN lies on separation of dataplane from the control programs providing flexibility to reconfigure the controller with change in network demands such as varying throughput, response time. On the other hand, it also introduces certain challenges towards scalability and performance, security hardening, cross-layer communication.

This work presents an efficient traffic scheduling architecture for SDN controllers. The basis of the scheduling architecture is driven by simulation based performance analysis of different state-of-art controllers. Our scheduling algorithm is designed over NOX, POX, Beacon and FloodLight controllers based on their response to different traffic. The scheduling decisions are taken after checking the contextual information from the OpenFlow packet header and the existing traffic load on the controller. The proposed solution can be realized in the hypervisor which can effectively schedule the traffic to different controllers.

On the other hand, we have also used multi-threaded execution of network functions in the controllers that provides significant enhancement in response time. The efficacy of our proposed packet scheduler is reported with experimental results that justify the effective inter-controller communication among various SDN domains.

Keywords: Software Defined Network (SDN) · Performance Controller · NOX · POX · Beacon · FloodLight · Packet scheduler Inter communication

1 Introduction

The recent trends of data digitization in large scale call for significant changes in the communication technologies and network platforms. In traditional network, computing infrastructure, and protocol stack may not be suitable to provide adequate solutions to such growing and heterogeneous demands. This leads towards a divergent approach in network systems architecture, called Software Defined Network (SDN). Software Defined Networking is a layered network framework

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 115–129, 2018.
https://doi.org/10.1007/978-3-319-92459-5_10

that provides unprecedented programmability, automation, and networks control by ramifying the control plane and the data plane of the network. In SDN architecture, network intelligence and states are logically centralized, and the underlying network infrastructure is abstracted for network applications. Network architectonics, where the control plane is separated from the data plane has been gaining popularity with scope of research and developments. One of the major features of this approach is that it provides a more structured software environment for developing network-wide abstractions while potentially simplifying the data plane. A generic architecture of SDN is shown in Fig. 1.

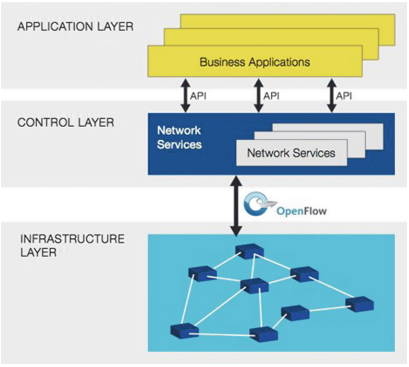


Fig. 1. Software Defined Network architecture

SDN offers various prevalence, such as centralized and decentralized control of multiple cross-vendor network elements, mainly data plane platforms with a common API abstraction layer for all SDN-enabled equipment. It dwindles the complexity of network configuration and operation that is achieved by automation of high level configuration and forwarding behaviour of network elements [7]. SDN acquiesces easy deployment of new protocols and network-services which leads towards high operation abstraction. SDN framework can adjust to the specific user application running on it through control plane, which abundantly improves the user experience [14].

Performance enhancement in SDN is a prerequisite for its usage in real life implications. Explicitly an initial estimation of the performance of openflow networks is crucial for network architects and designers [2]. The added flexibility and functionality require additional overhead on the equipment, and as a result there are performance forfeiture in terms of processing speed and throughput [5]. This is not suggesting that the overall performance is necessarily decreasing; many network services and tasks that were executed by the end-nodes or by the control layers of the network systems can be executed by the SDN-enabled equipment in a simpler and quicker way [8]. In addition to above mentioned issues, it also

leads to create a green environment, which we can call as Green ICT. Basically performance enhancement reduces energy consumption through out the network, mainly in data centers which leads to a green environment. SDN performance is evaluated considering following network parameters: Throughput, Latency, Jitter, Bandwidth, Response Time. In this paper, we evaluated the performance of SDN controllers based on the above mentioned parameters and reported the applicability of the various controllers in heterogeneous traffic scenarios. This comparison drives us to design a logical decision making module, called packet scheduler that schedules the traffic in different controllers with varying requirements. The scheduler can be implemented inside the SDN hypervisor. It has been observed that the performance of controller improves significantly based on our proposed controller architecture with appropriate inter SDN controller communication.

The rest of the paper is organized as follows. Section 2 presents the motivation and objectives of the work. We describe a study of different SDN controllers like NOX, POX, Beacon, Floodlight along with simulation results without our proposed packet scheduler algorithm in Sect. 3. Section 4 states about the packet scheduler inside the hypervisor, its functionality and its communication with the controller, also inter controller communication between various SDN domains. We conclude in Sect. 5 with further proceedings.

2 Motivation and Objectives

Now-a-days, large number of heterogeneous applications are being executed in the backbone networks of any organizations or publicly accessible networks [1]. On the other hand, the requirements of the organizations are becoming heterogeneous and stringent in terms of cost and QoS. In such scenarios, the increase in traffic and the varying requirements normally cause degradation in network's bandwidth, response time, throughput and SDN controllers' response time [19]. In order to maintain and enhance the performance of a network, the analysis of these parameters of SDN controllers in heterogeneous and live networks with varying network input is necessary. Analysis discloses relevant traffic for each controller. This analysis helps in bringing out a logical module which helps in forwarding the openflow packet to its appropriate controller.

The main objectives of this work are as follows:

1. To analyze the performance of SDN controllers, the network devices and to extract a comprehensive report with recommendations towards maintaining performance parameters.
2. To improve the existing network function execution and scheduling algorithms of controllers.
3. To design new algorithms for enhancement of SDN controllers performance and reduce energy, time consumption by the whole network and build a green environment.

3 Simulation of SDN Controllers

Controller is the crux of SDN, which regulates flow control to enable well-informed networking. It lies between network devices at one edge and applications at another edge. Any communication between applications and devices have to pass through the controller [12]. SDN controllers are hinged on protocols that concedes servers to reveal switches where to forward packets. Controllers make it possible to control the entire network from a single console. Here we considered 5 controllers with their working procedures, architectures, implementations and outcomes.

3.1 Overview of Controllers

(1) NOX: It was initially developed side-by-side with OpenFlow and was the first OpenFlow controller. It uses the concept of single threading and virtualization [20]. NOX is the basic level controller for performance enhancement. NOX's network view encompasses the switch level topology; the locations of users, hosts, middleboxes, and other network elements, and the services (e.g., HTTP or NFS) being granted. It also incorporates all bindings between names and addresses, but does not include the current state of network traffic. There exists multithreaded successor of NOX which is known as NOX-MT. NOX-MT uses well-known optimization techniques (e.g., I/O batching) to promote the threshold performance.

(2) NOX-MT: NOX-MT, a marginally modified multi- threaded successor of NOX, to show that with simple tease NOX's throughput and response time is significantly improved. The techniques used to optimize NOX are utterly well-known including: I/O batching to minimize the overhead of I/O, porting the I/O handling harness to Boost Asynchronous I/O (ASIO) library (which simplifies multi-threaded operation), and used a fast multiprocessor-aware malloc implementation that scales well in a multi-core machine [20]. It does not address many of NOX's performance deficiencies, including: heavy use of dynamic memory allocation and redundant memory copies on a per request basis. Addressing these issues would significantly improve NOX's performance.

(3) POX: POX is an OpenFlow networking software platform written in Python. POX started life as an OpenFlow controller, but now functions as an OpenFlow switch, and also be useful for writing networking software. POX provides OpenFlow interface and reusable components for path selection, topology discovery, etc. POX, which enables rapid development and prototyping, is becoming more commonly used than NOX.

(4) Beacon: Beacon is a fast, cross-platform, Java-based OpenFlow controller that supports both event-based and threaded operation [18]. Beacon runs on many platforms, from high end multi-core Linux servers to Android phones. Beacon architecture includes event handling, reading openflow messages, writing openflow messages to enhance performance.

(5) Floodlight: The Floodlight is an enterprise-class, Apache-licensed, Java-based OpenFlow Controller. It is supported by a community of developers including a number of engineers from Big Switch Networks. Floodlight can handle mixed OpenFlow and non-OpenFlow networks [17]. Floodlight is designed to work with the thriving number of switches, routers, virtual switches, and access points that support the OpenFlow standard. It peaks that the link between switch and controller is of primary importance for the gross performance of the network. If there is some problem in the link then directly it affects the performance of the controller. Also it cites that high latency is another cause of degradation of network performance [7].

3.2 Simulation Based Performance Analysis

For the purpose of performance analysis and evaluation, we have simulated four SDN controllers, namely, NOX, POX, Beacon, Floodlight and represented the results with necessary recommendations for SDN implementation and deployment in enterprise network. We have used following environments for our simulation.

Simulation Environment: Mininet 2.2.2.
SDN Controller Implementation: NOX, POX, Beacon, Floodlight.
Network Performance Tools: Cbench, iperf.
Operating System: Ubuntu 14.04 LTS-64 bit.

For network performance evaluation, we have done extensive simulation by varying network size from 5–100 nodes (hosts) with 10–20 OpenFlow switches, number of controllers are varying from 2–15. We also verified the following network parameters dynamically such as bandwidth, throughput, response time, latency and jitter. The throughput and latency of the controllers have been observed in both TCP and UDP mode using iPerf and Cbench tools. On the other hand, we calculated the network jitter in UDP mode. Each host generates heterogeneous traffic(both VBR and CBR) randomly. The performance variation in both client (host) and server (controller) systems have been observed. The simulation results are reported as comparison of throughput, latency and response time of the controllers. Figures 2, 3 and 4 show throughput, latency, packet drop ratio of 4 controllers respectively. POX provides better throughput and high latency than NOX. Beacon controller uses multi-threaded implementation along with event handling concept which is suitable for all types of traffic flows. Floodlight can handle both OpenFlow and non-OpenFlow traffic, so use of Floodlight controller brings maximum performance for the network. Overall findings of our simulation results are presented in Table 1, where all parameters values (no. of packets) are considered per second in average.

Table 1. Performance comparison of various controllers

Parameters	CBR traffic		VBR traffic	
	NOX	POX	Beacon	Floodlight
Throughput	1180	7569	30230	31615
Latency	1200	2100	3000	3200
PDR	13	8	4	2

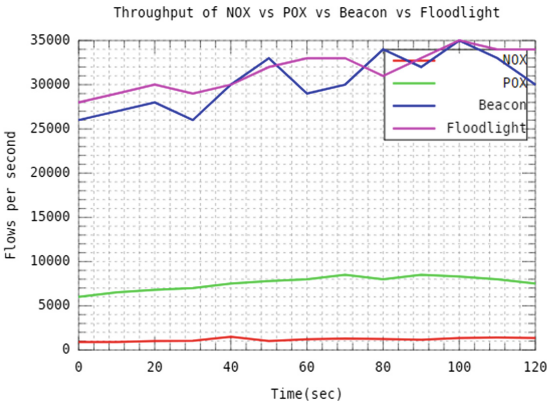


Fig. 2. Throughput comparison of controllers before introduction of packet scheduler

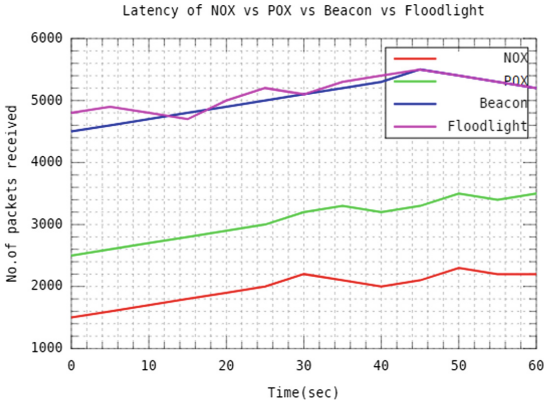


Fig. 3. Latency of controllers before introduction of packet scheduler

4 Maneuver of Packet Scheduler

This section describes the functionality of our proposed packet scheduler realized inside SDN hypervisor. Considering the results from previous section we designed a packet scheduler algorithm which schedules the network traffic

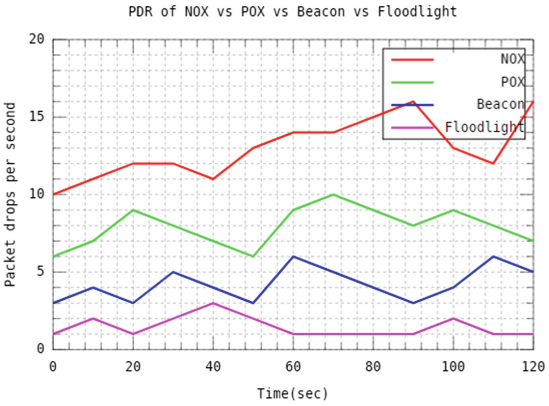


Fig. 4. PDR comparison of controllers before introduction of packet scheduler

efficiently and enhances the performance of the network than the existing ones. The proposed SDN architecture is shown in Fig. 5. The scheduler receives open flow packets from switches (Application Layer) and schedules the traffic to its appropriate controller. We discussed the performance of different controllers in the last section in terms of bandwidth, latency, jitter, throughput and response time. The scheduling algorithm is developed based on the performance result of the controllers. Whenever packet comes from the OpenFlow network, the module fetches features of incoming packets and forwards accordingly to the pertinent controller. It is important that, the presence of packet scheduler is nontransparent to the controller under consideration.

4.1 Implementation of Packet Scheduler

In this subsection, we describe the workflow of the packet scheduler.

First, it creates listener threads that checks for various events. The events are PacketIN, FlowMod, Error etc. Then, the listener thread forwards the packet to the specific segment that listens to the particular event. The packet scheduler consists of a Master Thread which maintains a queue of events as per their arrival.

In case of the event "new switch connection", the scheduler creates two subsequent threads and a new Socketchannel which performs READ and WRITE operations with the target controller. The SocketChannel is shared by both the threads and it is responsible for READ and WRITE operations from/to the switch and the target controller. The tuple of the packet along with the port is stored in the ControllerPort Class.

First Thread (Thread1) consumes packets and address of the target controller from the queue and sends it to the specified controller till the queue is not empty. Second thread (Thread2) reads data from the shared SocketChannel and writes data into the switch. Two threads are created for each new switch

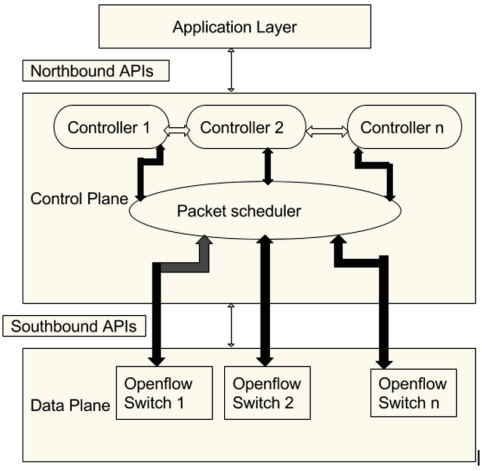


Fig. 5. Proposed SDN architecture

connections as shown in Fig. 6. The tuple of the packet along with the port is stored in the ControllerPort Class. First Thread (Thread1) consumes packets and address of the target controller from the queue and sends it to the specified controller till the queue is not empty. Second thread (Thread2) reads data from the shared SocketChannel and writes data into the switch. For creation of packet scheduler, we used multi-threaded implementation concept of Floodlight controller (Netty API for network I/O operations). The threads are terminated as the corresponding switch is disconnected.

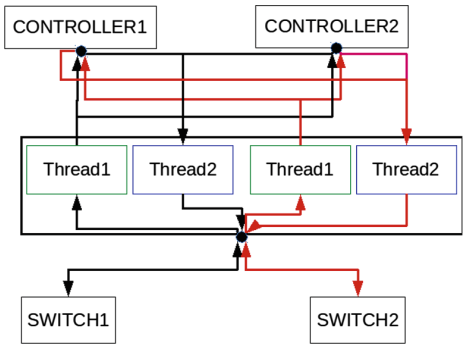


Fig. 6. Architecture of packet scheduler

The packet scheduler schedules the traffic depending upon following two metrics.

1. Load on the destined controller $L_{current}$ depends upon two network parameters, total message arrival rate (I) to the controller from switches and round trip time from each switch to controller (R). The round trip time (R) is measured by the number of hops present between controller and packet scheduler. If the load exceeds the predefined threshold (Thr) value, then the packet scheduler assumes that controller as heavily loaded and calculates load of other controllers present inside the network. The controller which is lightly loaded becomes the destination controller for that packet.

$$L_{current} = I \div R \tag{1}$$

2. Load on the path λ_{ij} is considered, depending upon link utilization U_{ij} and link capacity C_{ij} of each path exists between packet scheduler and the destination controller. Among all the paths, which ever is having less load, the packet will follow that path. Packet scheduler sets new destination port $D_{portnew}$ according to the traffic type and load of controller and path.

$$\lambda_{ij} = U_{ij} * C_{ij} \tag{2}$$

The pseudo code of packet scheduler is outlined in Algorithm 1.

Algorithm 1. Scheduling Algorithm of Packets

Procedure PacketSchedule ($packet(p_i)$)
Input : Stream of packtes p_i
Output: Processed Packets to the source port S_{port}
Read packets from the switch and store them in a queue
T_{diff}= arrival time difference between any two packets .
Thread List $TL = \Phi$
for *each packet* P_i **do**
    create threads T_1, T_2
    $TL = T_1 \cup T_2$
    Extract packet header information P_i<$S_{port}, D_{port}, length, data$>
    Calculate $L_{current} = I \div R$ and $\lambda_{ij} = U_{ij} * C_{ij}$
    if ($L_{current}$ <Threshold) then
    Set $D_{portnew}$ considering T_{diff}, $L_{current}$, λ_{ij}
    Thread T_1 : Connect ($D_{portnew}$);
    Send();
    Thread T_2 : Connect ($D_{portnew}$);
    Send(S_{port});
end
Return P_i<$S_{port}, data$>
end Procedure

4.2 Inter Controller Communication

One SDN controller manages one large scale network, termed as SDN domain. For greater scalability, security, performance, utilization of network resources efficiently, exchange of information, co-ordination of decision, multiple SDN controllers need to communicate with each other [15].

Requisite of Inter-SDN Controller Communication:

(1) *Isolated Controllers for various networks:* In order to fulfill customer demand, SDN controllers across different networks communicate among themselves and provide services with change in requirements. Occasionally requests come for data centers present in different networks, in these situations controllers from various networks communicate with each other.
(2) *Content delivery networks:* For particular content delivery requirement distinct SDN controllers communicate, in order to maintain load and serve the customer. Here is one example to explain this: a cricket match broadcast in India is being viewed by many fans. As more fans turn on their IP TVs or come on line to view the game, the load on the server increases, and the performance of the server decreases. In this situation, it is better to have customers get data from a server located elsewhere, let's in Europe, which has underused capacity.
(3) *Bandwidth on demand:* A network with single SDN controller, processes a request for bandwidth change instantaneously. However, when the network resources are distributed among multiple SDN domains, controllers from each domain have to communicate with the other SDN controller to share information on parameters like QoS, bandwidth availability, and so on. This enables the SDN controllers of each domain to confirm and process the bandwidth requirement. Without communication with SDN controllers of various domains, the processing of requests like bandwidth-on-demand will not possible.

Inter-SDN controller communication can be implemented in 2 ways: Vertical or Horizontal approach. Here for our implementation we considered the Horizontal approach. Horizontal approach uses the concept of peer-to-peer communication as shown in Fig. 7. Each controller communicates with its peer for connection establishment and information accession, that is SDN controller from other domain in the network.

Connection Between SDN Controllers: Implementation of multiple SDN domains leads to the matter of exchanging information between these domains. When SDN is deployed in a distributed large scale networks, our expectation is to visualize that deployment to a limited portion of the large networks. In other words, the large network is divided in to numerous SDN domains (a small portion of the large network), where each one is connected with another. These SDN domains are connected by means of the controllers present inside their regions. It leads to flexible provisioning of the networks, successive deployment and continuous evaluation, which extends to performance enhancement of the network.

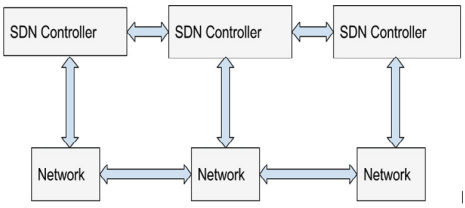


Fig. 7. Horizontal approach for inter-SDN controller communication

A session must be initiated between two controllers using either BGP (Border Gateway Protocol) or SIP (Session Initiation Protocol). Here we considered BGP connection over TCP for our better appliances [15]. Controllers need to exchange information such as:

(i) Reachability Update: During inter controller communication in SDN routing information need to be exchanged. This allows a single flow to traverse multiple SDN domains and each controller can select the most appropriate path in the network.
(ii) Flow setup and update Requests: Controllers exchange flow setup requests which contains path requirements, QoS among different SDN domains.
(iii) Capability Update: Controllers exchange network capability related information such as bandwidth, QoS, software and hardware capabilities available inside various SDN domains.

Steps involved in Inter controller communication is shown in Fig. 8. As the SDN controllers are in different SDN domains, they need to setup as BGP speakers. When the controllers are up, they triggered a BGP-Start event. The SDN controller establish a TCP connection with its neighbour. if there is problem in connection establishment (e.g. TCP time out or BGP message time expiry) BGP speaker has to establish connection with another neighbour. BGP uses TCP as its Transport layer protocol. Once the TCP connection is established, message sent by both controllers to each other. Once the BGP session is established, the BGP update messages are exchanged. Reachability data is exchanged between SDN controllers to expedite inter SDN routing. Bandwidth information is exchanged between SDN controllers through BGP UPDATE messages. QoS information is also carried in the BGP UPDATE message. As the two controllers are BGP speakers, the reachability data is preserved in the RIB table of each controller. All these information are converted into flows in the OpenFlow switches. Route selection depends upon BGP process decision when more than one path is accessible. Packets can traverse successfully once path is established between two SDN domains, and data exchange can take place through the BGP OPEN and UPDATE messages [15].

Throughput and latency of NOX, POX, Beacon, Floodlight controller are calculated after introducing the packet scheduler with inter SDN controller communication.

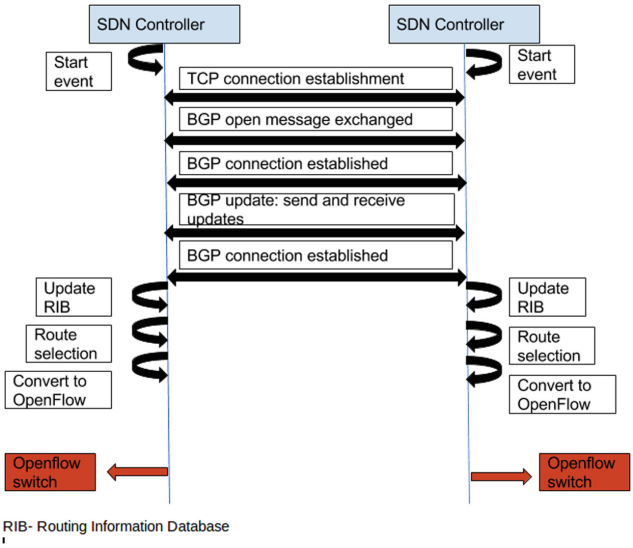


Fig. 8. Inter controller communication

The Figs. 9, 10 and 11 show throughput, latency and PDR comparison of NOX, POX, Beacon, FloodLight Controller respectively after introduction of proposed scheduling algorithm. The results show hike in performance, as throughput, latency value increased and also packet drop ratio (PDR) decreased with the packet. The comparison results of different network parameters with and with out packet scheduler is shown in Table 2, where all the values are presented as no. of packets per second in an average.

Based on the simulation results, We observed that the introduction of our proposed packet scheduler enhances performance of all controllers and also a polished communication takes place between different SDN domains. A higher increase in throughput and a decrease in PDR have been noticed in the network, which are clear indicative of controller's performance enhancement. Proposed packet scheduler is different from FlowVisor as it not only directs the network traffic but also calculates the controller and path load, which indicates better traffic management, load balancing with respect to performance enhancement of the whole network. Our proposed packet scheduler also reduces energy consumption by the network as well as takes less time for traffic management and flow, which has indirect impact on the environmental growth. Less energy consumption leads to less CO_2 emission and maintain a green environment, which can be expressed in terms of green ICT.

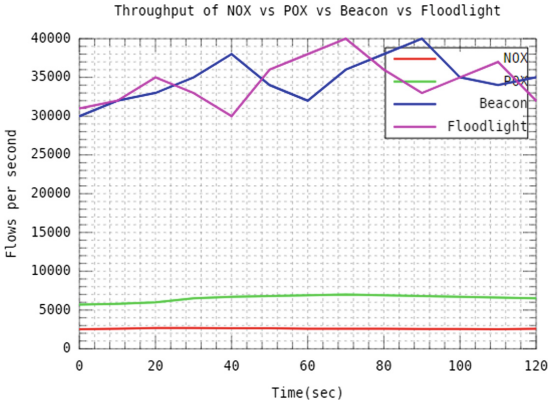


Fig. 9. Throughput comparison of controllers after introducing packet scheduler

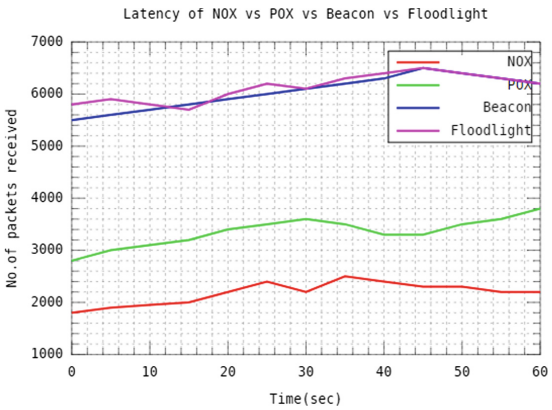


Fig. 10. Latency comparison of controllers after introducing packet scheduler

Table 2. Network parameter values of different controllers before and after introduction of packet scheduler

Network parameters	Before introduction of packet scheduler			After introduction of packet scheduler		
	Throughput	Latency	PDR	Throughput	Latency	PDR
Nox	1800	1650	15	3000	2000	7
POX	5000	2800	9	3100	5500	5
Beacon	30000	4300	6	38700	5800	3
FloodLight	33000	4800	4	40000	6000	2

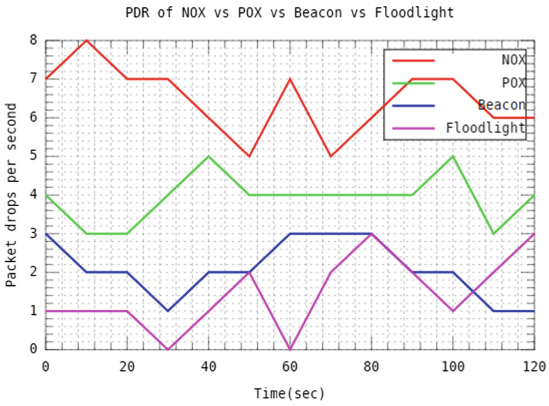


Fig. 11. PDR comparison of controllers after introduction of packet scheduler

5 Conclusion and Future Scope

The performance of controller is one of the major parameters in deploying SDN in real life implementation and applications. There is a need of developing novel SDN controller architecture towards serving a large amount of heterogeneous traffic with security guarantees. In this paper, we present a performance driven packet scheduler for SDN controllers. In one hand, the scheduler distributes load to different controllers and thereby achieves higher throughput and less latency. On the other hand, the scheduler directs the traffic to appropriate controller by checking the traffic type and parameters. If there is a need of inter controller communication, then it takes place through controllers of various SDN domains and thereby reducing the response time and jitter. We have observed significant performance enhancement in NOX, POX, Beacon and FloodLight controller.

The future scope of this work are as follows:

- We will work on the integration of packet scheduler with the Openflow switches towards achieving secure implementation and processing of the packets.
- In addition, we will work on developing controller algorithm and architectures for performance enhancement and ensuring end-to-end security of SDN controller.

References

1. Salman, O., Elhajj, I.H., Kayssi, A., Chehab, A.: SDN controllers: a comparative study. In: Proceedings of the 18th Mediterranean Electrotechnical Conference MELECON 2016, Limassol, Cyprus, 18–20 April 2016
2. Chen-xiao, C.: Research on load balance method in SDN. Int. J. Grid. Distrib. Comput. **9**(1), 25–36 (2016)

3. Rao, S.: A Guide for Running Multiple Controllers in Software Defined Networks, An Article on "TheNewStack", 21 March 2016
4. Bampal, R.: Cbench Data to Graph (2016). https://github.com/Rhnbmpl/cbench-data-graph
5. Bholebawa, I.Z., Jha, R.K., Dalal, U.D.: Performance analysis of proposed network architecture: OpenFlow vs. traditional network. Int. J. Comput. Sci. Inf. Secur. (IJCSIS) **14**(3), 943–958 (2016)
6. Hasan, H., et al.: Improvement of performance of EIGRP network by using a supervisory controller with smart congestion avoidance algorithm. In: International Conference on Telecommunications and Multimedia (TEMU) (2016)
7. Zhao, Y., Iannone, L., Riguidel, M.: On the performance of SDN controllers: a reality check. In: IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN) (2015)
8. Samociuk, D.: Secure communication between OpenFlow switches and controllers. In: The Seventh International Conference on Advances in Future Internet AFIN (2015)
9. Benamrane, F., Ben mamoun, M., Benaini, R.: Performances of OpenFlow-based software- defined networks: an overview. J. Netw. **10**(6), 329–337 (2015)
10. Ganesh, S., Ranjani S.: Dynamic load balancing using software defined networks. In: International Conference on Current Trends in Advanced Computing (ICCTAC) (2015). Int. J. Comput. Appl
11. Xia, W., Wen, Y., Foh, C.H., Niyato, D., Xie, H.: A survey on software-defined networking. IEEE Commun. Surv. Tutorials **17**(1), 27–51 (2015)
12. Khattak, Z.K., Awais, M., Iqbal, A.: Performance evaluation of OpenDaylight SDN controller. In: 20th IEEE International Conference on Parallel and Distributed Systems (ICPADS) (2014)
13. Shin, S., et al.: Rosemary: a Robust, secure, and high-performance network operating system. In: CCS 2014, Arizona, USA, 3 November 2014
14. Chilwan, A., et al.: On modeling controller-switch interaction in Openflow based SDN. Int. J. Comput. Netw. Commun. (IJCNC) **6**(6) (2014)
15. Jahan, R., Gupta, D.: White Paper Inter-SDN Controller Communication using BGP (2014). http://docplayer.net/5817317-Telecom-white-paper-inter-sdncontroller-communication-using-border-gateway-protocol.html. Accessed July 2017
16. Braun, W., Menth, M.: Software-defined networking using openflow: protocols, applications and architectural design choices. Future Internet **6**(2), 302–336 (2014). https://doi.org/10.3390/fi6020302
17. Shah, S.A., Faiz, J., M., Farooq, J., Shafi, A., Mehdi, S.A.: An architectural evaluation of SDN controllers. IEEE International Conference on Communications, pp. 3504–3508. IEEE (2013)
18. Erickson, D.: The Beacon OpenFlow controller. In: HotSDN 2013, Hong Kong, China, 16 August 2013
19. Gelberger, A., Yemini, N., Giladi, R.: Performance analysis of software defined networking (SDN). In: IEEE 21st International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (2013)
20. Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N., Shenker, S.: NOX: towards an operating system for networks. ACM SIGCOMM Comput. Commun. Rev. **38**(3), 105–110 (2008)

Quantum Coherence Measures for Quantum Switch

Marek Sawerwain[1(✉)] and Joanna Wiśniewska[2]

[1] Institute of Control and Computation Engineering, University of Zielona Góra, Licealna 9, 65-417 Zielona Góra, Poland
M.Sawerwain@issi.uz.zgora.pl

[2] Institute of Information Systems, Faculty of Cybernetics, Military University of Technology, Gen. W. Urbanowicza 2, 00-908 Warsaw, Poland
jwisniewska@wat.edu.pl

Abstract. We suppose that a structure working as a quantum switch will be a significant element of future networks realizing transmissions of quantum information. In this chapter we analyze a process of switch's operating – especially in systems with a noise presence. The noise is caused by a phenomenon of quantum decoherence, i.e. distorting of quantum states because of an environmental influence, and also by some imperfections of quantum gates' implementation. In the face of mentioned problems, the possibility of tracing the switch's behavior during its operating seems very important. To realize that we propose to utilize a Coherence measure which, as we present in this chapter, is sufficient to describe operating of the quantum switch and to verify correctness of this process. It should be also stressed that the value of Coherence measure may be estimated by a quantum circuit, designed especially for this purpose.

Keywords: Quantum information transfer · Quantum switch
Quantum coherence

1 Introduction

A field of quantum computing [23,24] called quantum communication [5,12,13,25] is a dynamically growing research area of network science [11,16]. Quantum communication deals, inter alia, with processing of quantum information. The concept of information transmission/switching in a quantum channel is one of the basic issues connected with quantum communication. In case of quantum information its transmission and switching are non-trivial problems because of non-cloning theorem [15,27] and decoherence [9].

A definition of quantum switch, realizing swapping information between channels A and B, was presented in [20]. A significant issue is to trace the operating of quantum switch, especially when distortions (caused by an influence of external environment) of quantum information occur. Correctness of the process

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 130–141, 2018.
https://doi.org/10.1007/978-3-319-92459-5_11

may be, for example, verified with use of quantum entanglement phenomenon – some basic information concerning this method was contained in [4,22]. In this chapter we suggest utilizing the Coherence measure [3] to evaluate the correctness of switch's operating. Furthermore, following the results described in [10] we propose of a quantum circuit that estimates of Coherence measure value for a quantum switch. It should be stressed that estimating the value of Coherence measure allows to trace the behavior of quantum system and clearly points out if it works properly.

The chapter is organized as follows. In Sect. 2 the basic information concerning the quantum switch was presented. This section contains description of the switch as a Hamiltonian and as a time-dependent unitary operation. There is also an example of distortions modeled with use of Dzyaloshinskii-Moriya interaction [7,19]. Sect. 3 contains definitions of Coherence measure, its properties and two examples of popular realizations.

A description of measures in context of the switch are presented in Sect. 4. We showed direct analytical formulas expressing Coherence measure for the switch during its operating. Conducted numerical experiments demonstrate changes in value of Coherence measure for systems with and without noise. We calculated also an analytical formula describing a difference between the values of Coherence measure for systems without presence of noise and with distortions modeled as Dzyaloshinskii-Moriya interaction.

The summary and plans for further work are presented in Sect. 5. A list of references to other works ends this chapter.

2 Quantum Switch

A quantum switch, analyzed in this chapter, is a system of three qubits denoted as: A, B, C. The states of qubits A and B are unknown:

$$|A\rangle = \alpha_A|0\rangle + \beta_A|1\rangle, \quad |B\rangle = \alpha_B|0\rangle + \beta_B|1\rangle. \tag{1}$$

The state of C is known and preserves only two possible alternatives: $|C\rangle = |0\rangle$ or $|C\rangle = |1\rangle$.

A main task of the quantum switch is swapping an unknown states between qubits A and B. This operation is performed conditionally: if the state of qubit $|C\rangle$ is $|0\rangle$ then there is no action; but if the state of qubit $|C\rangle$ is $|1\rangle$ then the values of quantum states are exchanged between qubits A and B:

$$|AB0\rangle \rightarrow |AB0\rangle, \quad |AB1\rangle \rightarrow |BA1\rangle. \tag{2}$$

The quantum switch may be depict as a circuit built of three quantum gates: two CNOT gates and one Toffoli gate. The conditional swapping of quantum states, realized by the mentioned circuit, is presented at Fig. 1.

Utilizing the circuit, shown at Fig. 1, we can denote a matrix form (also shown at Fig. 1 – case (c)) of an operator realizing the operation performed by the quantum switch describes the complete process of information switching

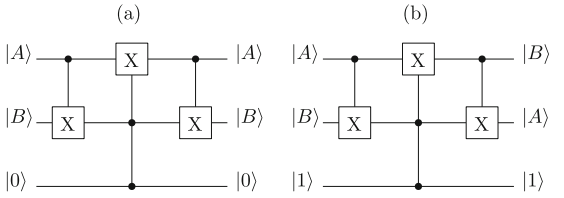


Fig. 1. An exemplary circuit presenting the action performed by the quantum switch. If the state of third qubit is $|0\rangle$ (case (a)) the switch does not swap the states of first two qubits. When the state of the last qubit is expressed as $|1\rangle$ (case (b)) the circuit swaps states $|A\rangle$ and $|B\rangle$. Case (c) depicts the matrix form of unitary operator which performs action of quantum switch

without details concerning particular steps of the process. Calculating a time-dependent unitary operation will allow to present the flow of information through the quantum switch during its operating time. To obtain this unitary operation we need to compute a Hamiltonian describing the dynamics of information flow in a quantum register:

$$H_{qs} = |011\rangle\langle 101| + |101\rangle\langle 011| = \begin{pmatrix} 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0 \end{pmatrix}. \tag{3}$$

The operator H_{qs} is Hermitian, so introducing a variable t we may obtain a time-dependent unitary operation:

$$U_{qs}(t) = e^{-\mathbb{i}tH_{qs}}, \hat{U}_{qs}(t) = \begin{pmatrix} 1&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&\mathbb{i}\cos(t)&0&\sin(t)&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&\sin(t)&0&\mathbb{i}\cos(t)&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&1 \end{pmatrix}. \tag{4}$$

The operation $U_{qs}(t)$ causes the change of local phase therefore using the phase-flip gate allows to obtain operator $\hat{U}_{qs}(t)$ which, naturally, does not introduce any change of local phase into the system.

Both, shown above, forms of unitary operation $U_{qs}(t)$ let to trace the process of switch's operating according to time t. If the switch acts on a state $|AB1\rangle$ and

there is no correction of the local phase then the final state is a pure quantum state:

$$U_{qs}(t)|\Psi_{qs}\rangle = |\Psi_{qs}^{t}\rangle = \begin{pmatrix} 0 \\ \alpha_0\alpha_1 \\ 0 \\ \cos(t)\alpha_0\beta_1 - \mathbb{i}\sin(t)\alpha_1\beta_0 \\ 0 \\ \cos(t)\alpha_1\beta_0 - \mathbb{i}\sin(t)\alpha_0\beta_1 \\ 0 \\ \beta_0\beta_1 \end{pmatrix}, \tag{5}$$

where $t \in \langle 0, \frac{\pi}{2}\rangle$.

And representation of this operation as a density matrix ρ is also given:

$$\rho(t) = U_q s(t)|\Psi_{qs}^{t}\rangle\langle\Psi_{qs}^{t}|U_{qs}^{\dagger}(t). \tag{6}$$

In Sect. 4 we are going to present the values of Coherence measure with the presence of noise. To do that the following Hamiltonian, describing Dzyaloshinskii-Moriya (DM) interaction [7,19], will be used:

$$H_{\mathrm{DM}} = D_z \cdot (\sigma_A^x \otimes \sigma_B^y - \sigma_A^y \otimes \sigma_B^x) \tag{7}$$

The notation of operator σ_A^x informs us that the Pauli operator X is used on the qubit A and similarly σ_B^y means that the Pauli operator Y is used on the qubit B. The symbol D_z represents the vector of process intensity. A Hamiltonian H_{TOT} joins the switch and DM interaction:

$$H_{\mathrm{TOT}} = t \cdot H_{qs} + D_z \cdot H_{\mathrm{DM}}, \quad U_{qs}^{\mathrm{DM}}(t, D_z) = e^{-\mathbb{i}(t \cdot H_{qs} + D_z \cdot H_{\mathrm{DM}})}. \tag{8}$$

where U_{qs}^{DM} stands for the unitary operation. It should be stressed that for D_z we obtain the Hamiltonian describing only the quantum switch's operating.

A state of quantum register including an influence of DM interaction may be expressed as:

$$U_{qs}^{\mathrm{DM}}(t)|\Psi_{qs}\rangle = |\Psi_{qs}^{U_{qs}^{\mathrm{DM}}(t)}\rangle = \begin{pmatrix} 0 \\ \alpha_0\alpha_1 \\ 0 \\ \frac{(2D_z - \mathbb{i}t)\sinh(\gamma)\alpha_1\beta_0}{\gamma} + \cosh(\gamma)\,\alpha_0\beta_1 \\ 0 \\ \cosh(\gamma)\,\alpha_1\beta_0 - \frac{(2D_z + \mathbb{i}t)\sinh(\gamma)\alpha_0\beta_1}{\gamma} \\ 0 \\ \beta_0\beta_1 \end{pmatrix}, \tag{9}$$

and $\gamma = \sqrt{-4D_z^2 - t^2}$. And also as a density matrix ρ:

$$\rho(t, D_z) = U(t, D_z)|\Psi_{qs}\rangle\langle\Psi_{qs}|U^{\dagger}(t, D_z). \tag{10}$$

3 Quantum Coherence Measures

Introducing the notion of Coherence needs to point the incoherent quantum states. This approach is similar to the measures of quantum entanglement where a set of separable states has to be specified. For a d-dimensional Hilbert space $\mathcal{H}$ we have to define a computational basis, for example the standard computational basis: $\{|i\rangle\}$, for $i = 0, 1, 2, 3, \ldots, d.$. The Coherence measure is basis-dependent, i.e. we consequently use the standard computational basis in this chapter.

Incoherent states $\mathcal{I}$ and maximally coherent state $|\phi_d\rangle$ are defined as:

$$\mathcal{I} = \{\delta | \delta = \sum_{d=0}^{d-1} \delta_i |i\rangle\langle i|\}, \quad |\phi_d\rangle = \frac{1}{\sqrt{d}} \sum_{i=0}^{d-1} e^{i\phi_i} |i\rangle, \tag{11}$$

where ϕ_i represents a phase of basis element $|i\rangle$.

Operations performed on states described as density matrices ρ are termed as Incoherent Operations (IO) if for a Complete Positive and Trace Preserving (CPTP) projection $\Lambda(\rho) = \sum_n K_n \rho K_n^\dagger$ we observe that $\frac{K_n \delta K_n^\dagger}{\mathrm{Tr}[K_n \delta K_n^\dagger]} \in \mathcal{I}$ for every value of n and $\delta \in \mathcal{I}$. A set of these operations may be denoted as $\Lambda(\delta) \in \mathcal{I}$ for each $\delta \in \mathcal{I}$ and it is called Maximal Incoherent Operations (MIO). Naturally: $IO \subseteq MIO$.

The Coherence measure for ρ is defined by a real-valued function $C(\rho)$. The function $C(\rho)$ has to fulfil the following properties:

(P1) $C(\rho) \geq 0, \forall \rho$ and $C(\delta) = 0$ if and only if $\delta \in I$,

(P2) monotonicity – the coherence measure C cannot increase its value if refers to an operation included in IO or MIO represented by channel Λ: $C(\Lambda(\rho)) \leq C(\rho)$,

(P3) strong monotonicity with post-selection – for any channel $\Lambda \in IO$, with given set of Kraus operators $\{K_n\}$, the coherence measure C cannot increase its average value under the post-selection:

$$\sum_n p_n C(\rho_n) \leq C(\rho), \text{ where } \rho_n = K_n \rho K_n^\dagger \mathrm{Tr}[K_n \rho K_n^\dagger],$$

(P4) convexity – the value of coherence measure C cannot be increased by mixing quantum states: $C(\sum_n p_n \rho_n) \leq \sum_n p_n C(\rho_n)$.

Generally, if the measure C fulfils the condition P1 and P2 or P3 then it is a monotone function and in this case the measure is very useful. Nowadays, the most widely use Coherence measures are: l_1-norm coherence (C_{l_1}) and relative entropy of coherence (C_{re}). These both measures fulfill the above-mentioned properties.

A significant advantage of both specified measures is their relative simplicity. The l_1-norm coherence is presented as a sum of all off-diagonal elements of density matrix and the relative entropy of coherence measure is defined as entropy difference:

$$C_{l_1}(\rho) = \sum_{\substack{i,j \\ i \neq j}} |\rho_{i,j}|, \quad C_{\mathrm{re}}(\rho) = S(\rho_{\mathrm{diag}}) - S(\rho). \tag{12}$$

where $S(\cdot)$ stands for von Neumann entropy and ρ_{diag} denotes a density matrix without the off-diagonal elements.

4 Coherence Measures for Quantum Switch

The Coherence measures will be utilized to assess the correctness of quantum switch operating. To do it we need to calculate values of these measures for two situations: when analyzed quantum state contains noise and without it.

If a quantum state is free from noise, we examine states of two first qubits A and B. After a partial trace operation, which aim is to remove a state of controlling qubit C, we obtain a reduced density matrix:

$$\text{Tr}_{\text{C}}\left(\rho_{ABC}\right) = \rho_{AB}, \tag{13}$$

where the states $|A\rangle$ and $|B\rangle$ are unknown. Utilizing measure (12) and carrying out some necessary transformations allows to calculate the following formula:

$$\begin{aligned} C_{l1}(\rho_{AB}) = 2\Big(& |\alpha_0\alpha_1\beta_0\beta_1| + |\alpha_0\alpha_1\left(\cos(t)\alpha_0\beta_1 - \mathbb{i}\sin(t)\alpha_1\beta_0\right)| \\ & + |\beta_0\beta_1\left(\cos(t)\alpha_0\beta_1 - \mathbb{i}\sin(t)\alpha_1\beta_0\right)| + \left(|\alpha_0\alpha_1| + |\beta_0\beta_1|\right. \\ & \left. + |\cos(t)\alpha_0\beta_1 - \mathbb{i}\sin(t)\alpha_1\beta_0|\right)|\cos(t)\alpha_1\beta_0 - \mathbb{i}\sin(t)\alpha_0\beta_1|\Big). \end{aligned} \tag{14}$$

A value of Coherence measure C_{l_1} depends on time and states of qubits. However, if the second qubit $|B\rangle = |0\rangle$, we can use (14) and compute:

$$C_{l1}(\rho_{A0}) = 2\,|\sin(t)\alpha_0\beta_0| + 2\,|\cos(t)\alpha_0\beta_0| + 2\left|\cos(t)\sin(t)\beta_0^2\right| \tag{15}$$

As we can see the value of this measure depends only on time elapsing during the switch's operating. Basing on the fact that the quantum state is normalized, we can directly indicate the constraint concerning the value of C_{l_1} measure for the state $|A0\rangle$:

$$C_{l1}(\rho_{A0}) < \left(\varepsilon + \left(|\sin(t)| + |\cos(t)| + |\cos(t)\sin(t)|\right)\right), \tag{16}$$

where the state $|A\rangle$ is unknown and the constant $\varepsilon \in \mathbb{R}$.

Generally, the changes of C_{l_1} value allow to trace the work of switch because at the beginning of operating, when $t = 0$, the value of C_{l_1} is minimal, then in the middle of the process, that is when $t = \pi/4$, the value of C_{l_1} is maximal. When the switch finishes operating in a moment $t = \pi/2$ the value of C_{l_1} is the same as in the moment $t = 0$. Figure 2 depicts the changes of C_{l_1} measure for quantum states:

$$|A\rangle = \frac{1}{\sqrt{10}}|0\rangle + \frac{\sqrt{9}}{\sqrt{10}}|1\rangle, |B\rangle = \frac{\sqrt{9}}{\sqrt{10}}|0\rangle + \frac{1}{\sqrt{10}}|1\rangle. \tag{17}$$

and, more generally, for the states described as:

$$|A\rangle = \sin(a)|0\rangle + \cos(a)|1\rangle, \quad |B\rangle = \sin(\frac{\pi}{2} - a)|0\rangle + \cos(\frac{\pi}{2} - a)|1\rangle \tag{18}$$

where $a \in \langle 0, \frac{\pi}{2}\rangle$.

Remark 1. If the time of switch's operating is extended, for example, to $t = \pi$ and the value of the parameter a is increased to π, we will observe a periodic character in changes of C_{l_1} value.

The C_{l_1} measure, utilized in quantum systems without presence of noise, offers also the direct possibility of checking if states A and B are the same – that is if the switch operates on state $|AA1\rangle$.

Theorem 1. *For the quantum switch defined as operation U_{qs}, like in (4), the value of C_{l_1} for state $|AA1\rangle$ remains unchanged in time t.*

Proof. The above theorem may be proofed by calculating the value of C_{l_1} for the state $|AA1\rangle$. After some transformations of Eq. (14) we obtain:

$$C_{l_1}(\rho_{AA}) = 4|\alpha_0\beta_0|\left(|\alpha_0|^2 + |\alpha_0\beta_0| + |\beta_0|^2\right). \tag{19}$$

A consequence described by Theorem 1 may be observed at Fig. 2 in case (b) where the top of the chart is flatten for $a = \pi/4$.

If the noise modelled by DM interaction is present in the system, the C_{l_1} measure shows the distortions' influence on a quantum state:

$$\begin{aligned} C_{l_1}^{\mathrm{DM}}(\rho_{AB}) = {} & 2\left|(\xi\alpha_1\beta_0 + \eta\alpha_0\beta_1)\left(\zeta\alpha_1\beta_0 + \xi\alpha_0\beta_1\right)\right| \\ & + 2\left|\alpha_0\alpha_1\left(\zeta\alpha_1\beta_0 + \xi\alpha_0\beta_1\right)\right| + 2\left|\beta_0\beta_1\left(\zeta\alpha_1\beta_0 + \xi\alpha_0\beta_1\right)\right| \\ & + 2\left|\alpha_0\alpha_1\left(\xi\alpha_1\beta_0 + \eta\alpha_0\beta_1\right)\right| + 2\left|\beta_0\beta_1\left(\xi\alpha_1\beta_0 + \eta\alpha_0\beta_1\right)\right| + 2\left|\alpha_0\alpha_1\beta_0\beta_1\right| \end{aligned} \tag{20}$$

where the following denotations were used:

$$\begin{aligned} \xi &= \frac{1}{2}e^{-\sqrt{-4D_z^2 - t^2}} + \frac{1}{2}e^{\sqrt{-4D_z^2 - t^2}} \\ \eta &= \frac{e^{\sqrt{-4D_z^2 - t^2}}(-2D_z - \mathbb{i}t)}{2\sqrt{-4D_z^2 - t^2}} - \frac{e^{-\sqrt{-4D_z^2 - t^2}}(-2D_z - \mathbb{i}t)}{2\sqrt{-4D_z^2 - t^2}} \\ \zeta &= \frac{e^{\sqrt{-4D_z^2 - t^2}}(2D_z - \mathbb{i}t)}{2\sqrt{-4D_z^2 - t^2}} - \frac{e^{-\sqrt{-4D_z^2 - t^2}}(2D_z - \mathbb{i}t)}{2\sqrt{-4D_z^2 - t^2}} \end{aligned}$$

We can observe that time t and the level of noise D_z directly affect a quantum state by modelling its amplitudes. Naturally, these modifications cause changes of the C_{l_1} measure value.

Utilizing the definitions of states given in (17) and (18) we present an exemplary process of C_{l_1} value's changes when the noise is generated by DM interaction, what is depicted at Fig. 2. The given characteristic was calculated for $D_z = 0.5$.

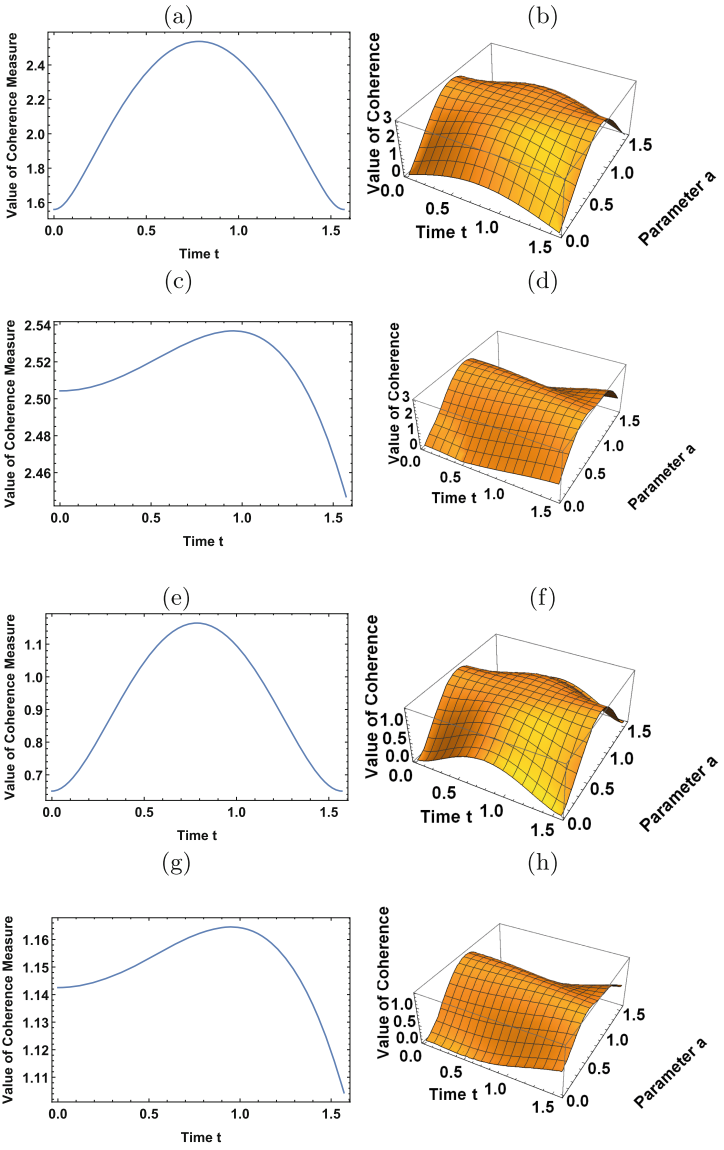


Fig. 2. The changes in value of C_{l_1} measure: (a) for some exemplary states A and B given in (17); (b) for a generalized case given in (18). Plots (c) and (d) present the changes in value of Coherence measure C_{l_1} when the noise, generated by DM interaction with intensity $D_z = 0.5$, is present: (c) for some exemplary states A and B given in (17); (d) for a generalized case given in (18). The changes in value of relative entropy measure C_{re}: (e) for some particular states; (f) for states A and B given in Eq. (18). Plots (g) and (h) show the changes in value of C_{re} measure when the noise, generated by DM interaction with intensity $D_z = 0.5$, is present: (g) for states given in (17); (h) for states A and B given in (18)

The properties (P1)–(P4) indicate that the values of relative entropy of coherence measure C_{re} also correctly depict the differences in a system's operating without and with presence of the DM interaction. For quantum states given in Eqs. (17) and (18) values of relative entropy measure C_{re} are shown at Fig. 2. Additionally, Fig. 2 demonstrates values of C_{re} measure with distortions generated by DM interaction.

Naturally, the Theorem 1 may be also based on relative entropy measure. It may be observed that the value of this measure is:

$$C_{\mathrm{re}}(\rho_{AA}) = -2\left(|\alpha_0|^2|\beta_0|^2\left(\log\left(|\alpha_0|^2|\beta_0|^2\right)\right)\right. \\ \left. +2|\alpha_0|^4(\log|\alpha_0|) + 2|\beta_0|^4(\log|\beta_0|)\right), \quad (21)$$

and again the value of time t is not used.

Regardless the measure, we can observe the difference between values of C_{l_1} in systems with and without noise. If the difference were always equal zero that implies the distortions are impossible to detect.

Proposition 1. *For the switch's states ρ_{AB} and ρ_{AB}^{DM} we denote the value of difference based on the C_{l_1} measure:*

$$C_{l_1}^{\Delta}(\rho_{AB}, \rho_{AB}^{DM}) = -2|\zeta\alpha_1\beta_0 + \xi\alpha_0\beta_1|\left(|\xi\alpha_1\beta_0 + \eta\alpha_0\beta_1| + |\alpha_0\alpha_1| + |\beta_0\beta_1|\right) \\ -2\left(|\alpha_0\alpha_1| + |\beta_0\beta_1|\right)|\xi\alpha_1\beta_0 + \eta\alpha_0\beta_1| + \frac{1}{2}\left|(\alpha_1\beta_0 + \alpha_0\beta_1)^2\right. \\ \left. -e^{4\mathrm{i}t}(\alpha_1\beta_0 - \alpha_0\beta_1)^2\right| + 2\left(|\alpha_0\alpha_1| + |\beta_0\beta_1|\right)\left(|\cos(t)\alpha_0\beta_1 - \mathrm{i}\sin(t)\alpha_1\beta_0|\right. \\ \left. +|\cos(t)\alpha_1\beta_0 - \mathrm{i}\sin(t)\alpha_0\beta_1|\right), \quad (22)$$

where ξ, η, ζ are denoted as in Eq. (20).

Utilizing Proposition 1 we can directly present an exemplary values of errors occurring during the switch's operating what is depict at Fig. 3. It should be stressed that the value of difference for exemplary plots at Fig. 3 equals zero only for two points of time t.

An analysis of works [6,8] shows that it is possible to design a quantum circuit for an estimation of Coherence measure value. Figure 3 depicts a circuit built to realize this task. The circuit takes an advantage on quantum states overlapping to estimate a value of Coherence measure – it is approximated with use of quadratic functional estimation of given density operator.

A significant task is to calculate the values of input states $|\alpha_{p_i}\rangle$ and to denominate the form of gate π_i, which usually is the SWAP gate. In [10] it was presented that using the circuit shown at Fig. 3 we can estimate the value of Coherence measure utilizing the Wigner-Yanase-Dyson skew information [26].

We would like to stress that the estimation of Coherence measure's value for states $|A\rangle$ and $|B\rangle$ may be calculated by a measurement, for example, of state $|\alpha_{p_0}\rangle$, what was shown in [8]. The probability that $|\alpha_{p_0}\rangle = |0\rangle$ may be estimated as:

$$\mathrm{Tr}\left(\rho_A\rho_B\right) = \pi_0 = 2P_0 - 1 \quad (23)$$

where $\rho_A = |A\rangle\langle A|$, $\rho_B = |B\rangle\langle B|$ and P_0 is measure projector, whereas π_0 represents estimated value of coherence. The value of this probability estimates also functionals values, including the values of Coherence measure. An accuracy evaluation of the circuit (Fig. 3) is under the analysis in currently prepared paper [21].

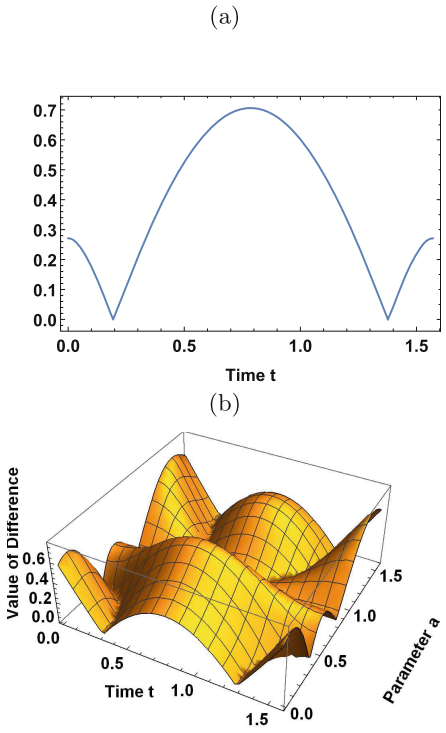


Fig. 3. The changes in value of absolute difference $C_{l_1}^{\Delta}$: (a) for states given in (17); (b) for states A and B given in (18). The parameter $D_z = 0.5$. Figure (c) represents a quantum circuit for estimation of Coherence measure

5 Conclusions

In this chapter we discussed utilizing the Coherence measure to trace and evaluate the correctness of quantum switch's operating. First, we described the switch with use of a Hamiltonian to be able to analyze the evolution of quantum system processed by the circuit shown at Fig. 1. We chose the Dzyaloshinskii-Moriya interaction as a source of noise, which was also modeled as a Hamiltonian. Then the numerical simulations were performed to evaluate a behavior of the circuit. As a tool to capture the differences between the switch operating, simulated

with noise or without it, two Coherence measures were used: l_1-norm coherence (C_{l_1}) and relative entropy of coherence (C_{re}).

We can observe the difference of C_{l_1} and C_{re} values if the system works with and without a noise. This shows that the Coherence measures allow to state if the switch operates properly. In addition, an interesting behavior of the system was observed when the switch worked on an initial state $|AA1\rangle$. The experiment showed that the value of C_{l_1} was constant during the simulation – for other states $|AB1\rangle$, where $A \neq B$, the value of measure evaluates in some characteristic periodic way. This implies that we can also check if the states are the same with use of C_{l_1} measure and quantum switch.

Unfortunately, we also observed that for the systems without and with distortions, modeled as DM interaction, the differences in Coherence measure value are significant even if their intensity D_z is quite low. It is inconvenient if we would like to capture the evolution of the system with the different levels of noise.

We presented a quantum circuit which estimates the value of Coherence measure. Nowadays, technical solutions based on quantum optics [1,2,17] and also physical implementations of qubits allow to build this kind of circuits with use of beam splitters, phase shifters and mirrors [14,18].

Acknowledgments. We would like to thank for useful discussions with the *Q-INFO* group at the Institute of Control and Computation Engineering (ISSI) of the University of Zielona Góra, Poland. We would like also to thank to anonymous referees for useful comments on the preliminary version of this chapter. The numerical results were done using the hardware and software available at the "GPU μ-Lab" located at the Institute of Control and Computation Engineering of the University of Zielona Góra, Poland.

References

1. Abubakar, M.Y., Jung, L.T., Foong, O.M.: Two channel quantum security modelling focusing on quantum key distribution technique. In: 5th International Conference on IT Convergence and Security (ICITCS), Kuala Lumpur, Malaysia, pp. 1–5 (2015)
2. Aguado, A., Lopez, V., Martinez-Mateo, J., Szyrkowiec, T., Autenrieth, A., Peev, M., Lopez, D., Martin, V.: Hybrid conventional and quantum security for software defined and virtualized networks. IEEE/OSA J. Opt. Commun. Netw. **9**(10), 819–825 (2017)
3. Baumgratz, T., Cramer, M., Plenio, M.B.: Quantifying coherence. Phys. Rev. Lett. **113**, 140401 (2014)
4. Bruß, D., Macchiavello, C.: Multipartite entanglement in quantum algorithms. Phys. Rev. A **83**, 052313 (2011)
5. Cariolaro, G.: Quantum Communications. Springer International Publishing, Heidelberg (2015)
6. D'Ariano, G.M., Perinotti, P.: Efficient universal programmable quantum measurements. Phys. Rev. Lett. **94**, 090401 (2005)
7. Dzyaloshinskii, I.: A thermodynamic theory of "weak" ferromagnetism of antiferromagnetics. J. Phys. Chem. Solids **4**(4), 241–255 (1958)

8. Ekert, A.K., Moura Alves, C., Oi, D.K.L.: Direct estimations of linear and nonlinear functionals of a quantum state. Phys. Rev. Lett. **88**, 217901 (2002)
9. Gawron, P., Klamka, J., Winiarczyk, R.: Noise effects in the quantum search algorithm from the viewpoint of computational complexity. Int. J. Appl. Math. Comput. Sci. **22**(2), 493–499 (2012)
10. Girolami, D.: Observable measure of quantum coherence in finite dimensional systems. Phys. Rev. Lett. **113**, 170401 (2014)
11. Goścień, R., Walkowiak, K.: A column generation technique for routing and spectrum allocation in cloud-ready survivable elastic optical networks. Int. J. Appl. Math. Comput. Sci. **27**(3), 591–603 (2017)
12. Imre, S., Gyongyosi, L.: Advanced Quantum Communications: An Engineering Approach. Wiley, Hoboken (2012)
13. Li, Y.H., Zhou, Z.Y., Xu, Z.H., Xu, L.X., Shi, B.S., Guo, G.C.: Multiplexed entangled photon sources for all fiber quantum networks. Phys. Rev. A **94**, 043810 (2016)
14. Lloyd, S.: Any nonlinear gate, with linear gates, suffices for computation. Phys. Lett. A **167**(3), 255–260 (1992)
15. Jozsa, R.: A stronger no-cloning theorem, arXiv: preprint quant-ph/0204153v2 (2002)
16. Markowski, M.: Heuristic algorithms for joint optimization of unicast and anycast traffic in elastic optical network-based large-scale computing systems. Int. J. Appl. Math. Comput. Sci. **27**(3), 605–622 (2017)
17. Mehic, M., Fazio, P., Voznak, M., Chromy, E.: Toward designing a quantum key distribution network simulation model. Adv. Electr. Electron. Eng. **14**(4), 413–420 (2016)
18. Milburn, G.J.: Quantum optical Fredkin gate. Phys. Rev. Lett. **62**(18), 2124–2127 (1989)
19. Moriya, T.: Anisotropic superexchange interaction and weak ferromagnetism. Phys. Rev. **120**(1), 91–98 (1960)
20. Ratan, R., Shukla, M.K., Oruc, A.Y. : Quantum switching networks with classical routing. In: 41st Annual Conference on Information Sciences and Systems, Baltimore, pp. 789–793 (2007)
21. Sawerwain, M., Wiśniewska, J.: Quantum circuit for estimation of quantum coherence during work of quantum switch, in preparation
22. Sawerwain, M., Wiśniewska, J.: The entanglement level and the detection of quantum data transfer correctness in short qutrit spin chains. Wirel. Pers. Commun. **96**(4), 5437–5452 (2017)
23. Węgrzyn, S., Bugajski, S., Klamka, J.: Foundation of quantum computing. Part 1 Archiwum Informatyki Teoretycznej i Stosowanej **1**(2), 97–142 (2001)
24. Węgrzyn, S., Bugajski, S., Klamka, J.: Foundation of quantum computing. Part 2 Archiwum Informatyki Teoretycznej i Stosowanej **14**(2), 93–106 (2002)
25. van Meter, R.: Quantum Networking. Wiley, Hoboken (2014)
26. Wigner, E.P., Yanase, M.M.: Information contents of distributions. Proc. Natl. Acad. Sci. USA **49**(6), 910–918 (1963)
27. Wootters, W.K., Zurek, W.H.: A single quantum cannot be cloned. Nature **299**, 802–803 (1982)

Performance Evaluation of Web Sites Using HTTP/1.1 and HTTP/2

Michał Druzgała[1] and Ziemowit Nowak[2(✉)]

[1] brightONE Sp. z o.o.,
ul. Piotra Skargi 3, 50-082 Wrocław, Poland
michal.druzgala@brightone.pl
[2] Faculty of Computer Science and Management,
Wrocław University of Science and Technology,
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland
ziemowit.nowak@pwr.edu.pl

Abstract. Introduction into the problem matter of the chapter. Overview of similar work (literature). Presentation of techniques of improving the performance of websites used in the HTTP/1.1 protocol and the problems they can generate in the context of the new possibilities of the HTTP/2 protocol. Description of the research stand and experiments. Discussion of the results of measurements of the speed of downloading websites using various techniques. Formulation of conclusions. The directions for further research were pinpointed.

Keywords: Domain sharding · Resource inlining · Concatenation
Minification

1 Introduction

When designing a website, one can influence how the site will be processed on the target user's side through a properly designed structure of HTML, CSS or JavaScript code, as well as by the proper preparation of the resources that make up the site in terms of structure as well as location. As a result, an important aspect of website's design is its performance.

When designing an efficient website, i.e. one that will be quickly processed and displayed on the client's side, it is possible to influence its reception, the general feeling of smoothness and convenience of using the website. A number of techniques, or proven methods are helpful here that in a universal way fit into the concept of shortening the loading times. They are based, inter alia, on the knowledge of the principles of the HTTP protocol, its limitations and disadvantages [1].

Currently, however, the use of some techniques is questionable in view of the increasing share of the new version of the HTTP – HTTP/2 protocol, which offers improvements resulting from the growing demand for efficient websites [2].

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 142–157, 2018.
https://doi.org/10.1007/978-3-319-92459-5_12

Due to the lack of a full understanding of the impact of the techniques used so far on HTTP/2 performance, there appeared a need to analyse the classic rules developed for the HTTP/1.1 protocol, in terms of their further relevance in view of the prospect of a significant increase in the new version of the protocol.

2 Related Works

Liu et al. [3] conducted a study of the effectiveness of the HTTP/2 protocol in terms of displaying a site on a mobile device. Using copies of the 200 most popular sites according to the Alexa rankings on the local server, they conducted a comparative analysis of HTTPS and HTTP/2 protocols for the impact of factors such as Round-Trip Time (RTT), bandwidth, packet loss rate and size of resources. The research showed the advantage of the HTTP/2 protocol, among others in situations when there was a need to download many smaller objects and the low rate of packet loss. The authors also noted the possible undesirable effects associated with the use of good practices derived from the previous version of the protocol, such as domain sharding, concatenation or inlining.

Corbel et al. [4] examined Page Load Time (PLT) using an unencrypted version of the HTTP/1.1 and HTTP/2 protocols for the prepared "average site", consisting of various types of resources stored on several different domains. The measurement scenarios assumed several separate delay profiles and the loss rate of the packet. The analysis showed the advantage of the HTTP/2 protocol in each scenario. The authors stressed the need to extend the research with the new mechanisms available in the HTTP/2 protocol and scenarios taking into account the size of the TCP window at the client's.

de Saxc'e et al. [5] analysed the impact of the HTTP/1.1 protocol change on the HTTP/2 protocol in the context of web site load performance. The 20 most popular sites according to the Alexa rankings were used for the purposes of the research and they were adapted to the needs of the local measurement environment and the environment similar to network realities (a dedicated HTTP server located outside the local network). PLT was determined as a measure of the loading performance of the websites. Measuring scenarios took into account various pre-defined network conditions (including ADSL and 3G). The research consisted of a number of scenarios using the new HTTP/2 protocol: compression, multiplexing, server push and prioritization. As a result of the research, the authors found the advantage of the new version of the protocol. They noted, however, worse performance in 3G network conditions (significant vulnerability to packet loss).

Varvello et al. [6] developed a measurement platform to track the use of the HTTP/2 protocol among the million most popular websites according to the Alexa ranking. Their research showed, inter alia, that the owners of most sites that adopted the new version of the protocol retained techniques developed for the purposes of the HTTP/1.1 protocol. Among those inlining or domain sharding can be mentioned. About 80% of registered sites supporting the new version of the protocol obtained better PLT. The best results were recorded for 3G networks in Europe.

Zarifis et al. [7] performed PLT analysis after switching to the new version of the HTTP protocol. For research, they used data from the Resource Timing API for approximately 280,000 visits by Akamai CDN users. Based on the modelling of the HTTP/2 server behaviour, using the collected visits measurements for Akamai servers (HTTP/1.1), they tried to analyse the theoretical decreases in PLT times. The authors showed that theoretically, the use of new techniques offered by HTTP/2 (server push, prioritization) would have a positive impact on the final loading times of websites.

Stepniak and Nowak [8] examined the impact of the new version of the protocol on web applications of the Single Page type (SPA) using different types of techniques to accelerate the loading of the site (concatenation, compression, minification). In the context of the HTTP/2 protocol use, they also examined the capabilities of the server push technique. The research showed a significant advantage of the HTTP/2 protocol in the event of having to download many smaller resources. The only aspect that had a negative impact on the loading times of the site was the use of the server push mechanism.

Rosen et al. [9] analysed the potential benefits of using the server push technique available for the new version of HTTP from the mobile clients' perspective. Their research involved 50 sites from among the top 500 of the most popular according to Alexa's ranking, adapted to the local conditions. As a result, they found measurable benefits stemming from the use of the new technique. Still they pointed to reduced efficiency of the technique for sites that use multiple servers to store resources, and in the case of a scenario involving a mobile client with a device inferior in performance, which may negatively affect the final processing time of the site (longer download times of the HTML file).

Kim et al. [10] carried out research on the efficiency of loading sites that use multiple domains to store resources in the context of the HTTP/2 protocol. They performed PLT measurements for typical Korean sites (which usually use multiple domains in their structure), copied to the local machine to be analysed. To construct the research environment, they used a server natively supporting the HTTP/2 protocol. The research showed increased PLT in all the adopted scenarios.

Seemakhupt and Piromsopa [11] analyzed scenarios in which the use of stream multiplexing using the HTTP/2 protocol brings the greatest benefits and may lead to a deterioration in site loading efficiency. They adopted two scenarios in the research. In the first one, the client individually queried the HTTP server for the resources of the site. In the other, an additional client was burdening the link. The results showed that the use of the HTTP/2 protocol using multiplexing performs better under low load conditions. However, under load conditions, the new protocol version performs worse than the HTTP/1.1 protocol.

Prokopiuk and Nowak [12] examined the impact of the new version of HTTP on the performance of the site from the perspective of the end user (User-Perceived Web Application Performance). They used WebPagetest tool to measure the performance of the examined websites, installed on the local client machine. The tool allowed them to acquire a number of measurements that are

relevant to the user displaying the site. Network conditions were simulated by the authors with the help of the Dummynet tool. As a result of the research, the authors found a beneficial effect of the server push mechanism in almost every scenario. The greatest benefit came from attaching CSS files and a background image.

3 Review of Issues Related to the Use of Old Techniques for the New Protocol

With the introduction of the new version of the protocol, some of the techniques used for the HTTP/1.1 protocol no longer bring time benefits or – even worse – negatively affect the total loading times of sites. This section presents problems resulting from preserving the old solutions after switching to the new version of the HTTP protocol and suggestions of the new approaches when designing sites using the HTTP/2 protocol [1,2].

3.1 Domain Sharding

With the development of websites, the need for parallel processing of resources increased. In the context of the HTTP/1.1 protocol, this meant using additional TCP connections, even within the same domain. Due to the overhead generated by additional connections, browser developers established internal limits of maximum connections within the domain in order to eliminate the risk of possible overloads of the system. For example, for Google Chrome, the maximum connection limit is 6. In many cases, the connection limits were too low, which motivated the creation of a new technique - domain sharding.

The idea of the technique is simple. In order to "cheat" the browser, some resources are shown in another, additionally created domain, which in fact points to the same server. Thanks to the use of the technique, it is possible to exceed the set limits and increase the parallel processing of resources in a browser-independent manner.

Formulating the Problem. From the perspective of the new mechanism and message exchange structure for the HTTP/2 protocol, technical improvements cease to be improvements. The HTTP/2 protocol natively supports full parallelization and multiplexing of messages within a single TCP connection.

The use of domain sharding for the HTTP/2 protocol generates overhead associated with, among other things, the creation of new TCP connections, while there is no need to do so. It is not possible either to use the full potential of mechanisms for prioritizing and controlling the flow of messages due to the additional TCP connections created. To some extent, the header compression mechanism also suffers, as it must operate in this situation separately for each connection, introducing a certain degree of redundancy.

Suggested Solution. Domain sharding is the first technique whose discontinuation should be strictly considered. It avoids constraints that no longer exist in the new version of the protocol (the problem of new TCP connections and the limited parallelism of resource processing).

Direct cancellation of the technique shall have a positive impact on the site's performance, which after the change will use the minimum number of TCP connections. In addition, it should be taken into account that in order to effectively implement the prioritization and flow control mechanism, transmission of flows within the minimum number of TCP connections is required. An important advantage is also the correct operation of the HTTP header compression mechanism (the header context is remembered within a single TCP connection).

3.2 Concatenation of Resources

In 2007, the total processing time of the site was only 10÷20%, which included the time to download the HTML document itself [13]. The remaining time was devoted to the processing of additional resources appearing as references in the HTML structure. Due to the growing number of HTTP requests, the overhead associated with performing an increased number of resource queries became more and more important from the performance perspective.

A technique of concatenation, i.e. joining resources of one kind in order to reduce the number of queries, came to the rescue. Merged resources that have been downloaded must be logically processed to reach their condition from before the connection. For merged images (sprites) CSS styles are created that "cut" the one that is needed on the site. There is no need for additional steps for CSS and JavaScript resources.

Merged resources are readable by the browser in this form. The use of the concatenation technique is able to reduce the loading times of sites by up to 50% [13].

Formulating the Problem. In the context of the HTTP/1.1 protocol, the use of concatenation is beneficial by reducing the problem of simultaneous processing of many smaller resources. The HTTP/2 protocol reveals its full potential in the scenario of high resource granulation. The mechanism of multiplexing streams within a single TCP connection is able to handle single resources in the case of parallel processing.

The use of the concatenation technique in the context of the HTTP/2 protocol creates the problem of a longer wait for a specific resource due to its connection with others. Another issue is cache management. The smallest change in the merged resource forces makes it necessary to reprocess the whole (redundancy of data transfer).

Suggested Solution. The concatenation technique is designed, among other things, to reduce separate requests to the server, which has a positive impact on performance in the context of the HTTP/1.1 protocol. Another effect of applying

the technique is also an increase in compression efficiency for related resources, which is a positive aspect also in the new version of the protocol.

Due to the above, this technique should not be completely abandoned, but should not be overused either. One should look for the "golden mean".

3.3 Resource Inlining

The resource inlining technique allows one to achieve the effect of directly downloading resources along with an HTML document in a single HTTP response. The selected resources are "embedded" (inlined) directly in the HTML document, forcing the transfer of the resource along with the structure of the site as a result.

Embedding a resource may look different depending on the type of resource. In the case of CSS styles or JavaScript scripts, it is enough to place the content directly in the HTML structure between relevant tags. In the case of multimedia, it is possible to attach encoded content in Base64 directly in the HTML tag.

This is another way to reduce the number of queries to the server and in a way to prioritize some resources (the embedded resources naturally receive the highest priority). Increased loading times of the base HTML structure should be borne in mind, however, which delay the start of sending requests for external resources.

Formulating the Problem. The resource inlining technique offers measurable benefits in the context of the HTTP/1.1 protocol due to its contribution to solving the problem of parallel processing of many resources and the reduction of excess RTT times caused by the processing of separate resources.

The different characteristics of the HTTP/2 protocol mean that the approach has a negative impact on the website's loading performance due to a breach of the concept of multiplexing and prioritizing the transfer of resources that must be separately identified in the structure of the HTML document. There is also the problem of data duplication in the case of multiple use of the same resource. As well as the problem with the effective use of the cache control mechanism.

Suggested Solution. The new HTTP/2 mechanism, server push, provides an alternative to resource inlining in terms of query reduction. Even the most naive strategy adopted by the server is better than the direct embedment of the resource, over the downloading of which the client has no control.

3.4 Minification

Another aspect of shortening transmission times is the reduction of the direct content of the transferred resource. In the case of resources containing CSS or JavaScript code, this can be achieved using a technique called minification (and obfuscation).

Minification involves reducing the white characters used in the code to improve its readability in the course of its development. White characters are reduced to a minimum while maintaining the semantic correctness of a given code. As a result, the resource should not differ from its predecessor in terms of functionality. Such a modified resource should be included in the document in order to send a request for an already identified resource (one-off overhead related to the preparation of the resource).

Obfuscation (darkening) goes a step further, modifying to the minimum the names given to all programming entities (variables, functions and others) while maintaining syntactic and semantic correctness with the original. The yield in relation to the version of the modified resource is noticeable [13].

The experiment [13] carried out on the modified and obscured JavaScript resources showed a time gain of $17 \div 31\%$ (depending on the size of the resource).

4 Analysis of the Speed of Websites Operation Using Various Techniques

4.1 Research Assumptions

When analyzing the impact of changing the protocol on the performance of the site, the main measure of performance was the size of PLT [14], the time measured between the start of navigation to the site and the moment indicating the completion of processing all additional resources associated with the examined site.

The user perceived performance aspects were not subject to the study, which means that information about the speed and order of rendering resources by a particular browser was not collected and analysed.

The following profiles of the clients visiting the website were adopted:

- desktop client with Google Chrome 59,
- desktop client with Mozilla Firefox 53 browser,
- mobile client (Android) with Google Chrome 59.

As a source of data on the speed of loading the site, only the target client's site (supported browser) was determined.

Bearing in mind the assumptions regarding the site's performance measurement, the decision was made to use the Resource Timing API. It is a JavaScript programming interface that enables one to obtain precise, high-resolution time information (parts of milliseconds) for all resources that are part of the site. Through this interface, it is possible to access information about all HTTP request components about the resource [15,16].

Jetty 9.4 server was chosen to handle HTTP/2 requests. For the proper functioning of the server, it was required to use an additional ALPN (Application-Layer Protocol Negotiation) library, allowing the negotiation of the protocol used at the application layer.

As a basis for the preparation of websites in the simulation environment, a partially configured Spring Boot demo project with the Jetty server was used

[17]. The project was modified to support multiple ports, implementation of the server push mechanism for explicitly defined resources that were part of the selected site and implementation of the mechanism for capturing and recording measurement results.

In order to examine the performance of sites using each protocol individually, a solution was chosen involving the selection of the protocol version by using appropriately assigned ports:

- port 8080: using the HTTP/1.1 protocol,
- port 8443: using the HTTP/2 protocol.

In both cases, connection encryption was used.

Taking into account the variability of network conditions on a larger scale over time, the decision was made to conduct measurements in an isolated local area network where only one server with the examined site was located at the time of measurement, and one client querying server resources.

Being aware of the fact that it is the first visit that is the key from the perspective of the user visiting the site, the decision was made to focus on analysing only the first visit to the site. It means that all techniques that primarily use the cache mechanism will not work in a simulation environment, so they were not taken into account. In order to simulate the continuous effect of the first visits, it was decided to use developer tools in the browsers, and more specifically, the option to disable the cache.

In an isolated environment, reflecting the reality of conditions in terms of bandwidth and prevailing delays while maintaining these conditions unchanged, a bandwidth control and server-side latency mechanism was adopted. Taking into account the possible discrepancy in the effectiveness of the techniques used in terms of network conditions, the following two network profiles were determined subject to separate measurements:

- DSL profile: 2 Mbit/s bandwidth, 5 ms delay,
- 3G profile: 750 Kbit/s bandwidth, 100 ms delay.

Dummynet [18] was used to simulate the established profiles. It is a tool that allows one to emulate specific network conditions in real time. Originally created for FreeBSD, it is now also available for Linux, OSX and Windows distributions. The tool uses ipfw firewall software. Ipfw allows communicating with Dummynet via the command line and setting traffic restrictions that ultimately are ipfw firewall rules. Installed as a service on a network card in Windows, it is able to capture traffic and subject it to various modifications, including bandwidth and delays.

4.2 Research Stand

In order to meet the research assumptions and to ensure precise research results, a dedicated simulation environment was developed (Fig. 1).

The HTTP server was a PC computer with the following parameters:

– Intel Core i5 processor,
– 8 GB RAM,
– Windows 7 32-bit operating system.

The server software was installed on the computer along with the sites to be examined. On its network card, the Dummynet and ipfw drivers were installed and, as a result, it was possible to properly manipulate network conditions from the level of the desktop console.

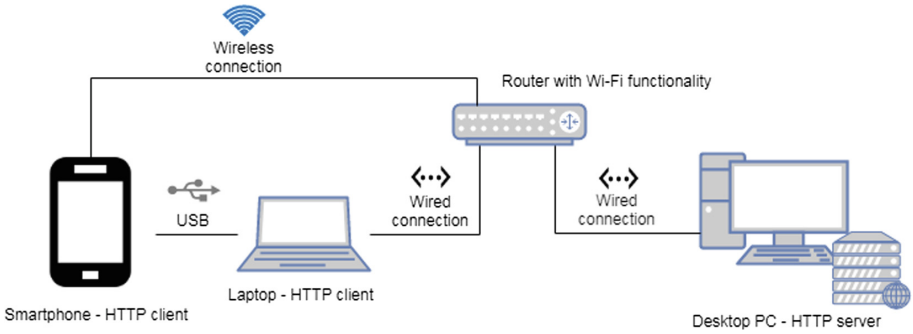


Fig. 1. Schematic drawing of the research stand.

The HTTP client was based on the Dell Latitude E6440 laptop with the following parameters:

– Intel Core i5 processor,
– 8 GB RAM,
– Windows 7 32-bit operating system.

On the HTTP client laptop, Google Chrome and Mozilla Firefox were installed. In both browsers, developer tools and private mode were launched and activated.

The OnePlus One smartphone was used to examine the mobile version of Google Chrome using the following parameters:

– Qualcomm Snapdragon 801 processor,
– 3 GB of RAM,
– Android 6 (Marshmallow) operating system.

During the research, the smartphone-client HTTP connected to the router using Wi-Fi technology in the 802.11n standard.

4.3 Sites Examined

According to the adopted assumption regarding the separation of the applied techniques, in order to carry out measurements for each technique individually, it was necessary to construct separate sites for each technique separately with the division into the "raw" version (without using any technique) and the version using the chosen technique. To minimize the need for additional queries, scripts analysing website load times and any CSS style sheet sources were placed directly in the HTML code.

Concatenation. The main part of the page under investigation for the technique of concatenation were six resources in the form of images in the PNG format. As a raw state, six separate resources defined in the HTML code were determined for the purpose of rendering six images, which involved their complete separation. As a result of such defining the structure of the site, it was necessary to download each resource separately.

After applying the concatenation technique, the resources were merged into one larger resource, which was later logically separated on the client's side into individual resources using the CSS styles attached to the HTML structure.

Due to the concatenation technique, only one graphic resource was referenced, and then its logical "cutting" into individual images took place. As a result, the HTML object code rendered the images by appropriate references to the CSS classes.

Inlining. Six resources in the form of images in PNG format were reused for the purposes of the inlining technique. Also, the "raw" state was determined by the need to download all other additional resources.

Following the application the inlining technique, resources became an integral part of the HTML document. References to additional resources were replaced with references to resources encoded directly in the document using Base64 encoding.

With the "raw" version of the site, a solution utilizing the server push mechanism, available in the new version of the protocol, was also analysed. Due to the architecture of the measurement system, an additional version of the site was created for the purpose of proper site selection on the backend side. On the side of the HTML document, the structure did not change with respect to the "raw" version. Additional graphic resources still had to be downloaded. The difference was due to the use of the server push mechanism on the backend side using the explicit resource selection.

Domain Sharding. A separate site was prepared for the domain sharding research purposes. It contained in its body references to twenty graphics resources that constituted components of a certain graphics composition.

Queries to additional resources were made in a standard way, by including HTML graphics elements. As a result of establishing such structure of the HTML document, it was necessary to query the server for each element separately.

When using the HTTP/1.1 protocol, the limits of simultaneous connections within a single domain had a significant impact on the performance of such a solution. Thus, additional "virtual" domains were used, entered for the purposes of research into the hosts file. Instead of twenty queries for one domain, queries were divided into four additional domains, allocating a maximum of six resources per domain. Behind the additional domains there was exactly the same IP address from which the HTML document was downloaded.

Due to the need to fully use the HTTP/2 protocol, a third version of the site was created, having the same structure, differing only in the port number (8443 instead of 8080), used for queries for additional graphics resources.

Minification. For the purpose of analysing the effectiveness of the minification technique, an appropriate website was also prepared. In the raw document structure there were three references to additional resources in the form of JavaScript libraries and CSS libraries with a significant size in relation to the rest of the site's resources. In the version using the minification technique, three references to additional resources were also used, but this time their internal structure was changed to reduce their size.

4.4 Methodology

Every possible scenario of the user's visit was subject to a separate measurement series. The scenario should be understood as a set of the following factors:

- network profile,
- application of the techniques (or the lack thereof),
- protocol version,
- type of browser (along with the type of HTTP client device).

The measurement series consisted of twenty visits, which were individually recorded in the database as one measurement. Each subsequent visit to the resource page, being an element of a single measurement, proceeded in a sequential manner. Each subsequent measurement was started only at the end of the previous one. Between each individual measurement, a time window of one second was established.

The next step was the data cleansing phase, which consisted of the following activities:

- removal of outliers and measurements with a gross error,
- verification of areas of component values of measurements,
- verification of the correctness of writing to the database.

After cleaning the collected data, each case was analysed in detail. Histograms of purified data were examined to establish the distribution, which directly affected the decision regarding the adopted measurements.

As a result of the analysis, it was decided that the median values from the measurement series would be used as the measure of central tendency for the representation of the results of individual scenarios. The use of median is justified due to the fact that most of the histograms of the measurement series for the determined scenarios had a skewed distribution.

4.5 Results of the Analysis and Conclusions

The lists of average PLT times are presented in four tables. In the case of scenarios that differ from the "raw" scenario for the HTTP/1.1 protocol, the percentage gain (PLT) of the PLT time in relation to this scenario is also given.

Taking into account the current trends in the context of users' preferences regarding web browsers and the continuous increase in the share of mobile clients, after a thorough analysis of the interpreted results, the following conclusions can be drawn regarding the use of selected techniques in conditions that take into account the potential use of HTTP/2 for sites with similar of character in relation to the examined websites.

Concatenation. When deciding on changes in the context of applying the concatenation technique, several factors should be taken into account (Table 1). The trends regarding the website's clients in terms of used browsers and platforms (mobile/desktop) should be examined.

Table 1. Comparison of Page Load Time for **concatenation** scenarios.

Concatenation		HTTP/1.1			HTTP/2.0			
		raw	active		raw		active	
		PLT [ms]	PLT [ms]	Gain [%]	PLT [ms]	Gain [%]	PLT [ms]	Gain [%]
DSL	Chrome PC	148.0	152.0	−2.70%	141.0	4.73%	149.0	−0.68%
	Firefox	232.0	184.0	20.69%	192.5	17.03%	181.0	21.98%
	Chrome mobile	272.0	276.0	−1.47%	267.5	1.65%	268.5	1.29%
3G	Chrome PC	453.0	446.5	1.43%	647.0	−42.83%	647.0	−42.83%
	Firefox	1893.0	1278.0	32.49%	1489.0	21.34%	1605.5	15.19%
	Chrome mobile	636.0	613.5	3.54%	757.0	−19.03%	769.5	−20.99%

If the majority of users of the site are desktop users with a constant Internet connection using the Mozilla Firefox browser, it is recommended to continue using the concatenation technique (17.03% gain PLT for the "raw" version and 21.98% gain for the "active" version).

Otherwise, what is meant by a larger share of mobile clients or using the Google Chrome browser, it is discouraged to continue using the technique due to

better PLT results without using the technique in the context of the new version of the protocol (higher gain of the "raw" version compared to the "active" version by up to 6.15 pp).

Inlining. In the inlining technique, the current trends may also influence the decision regarding its further use (Table 2).

Table 2. Comparison of Page Load Time for **inlining** scenarios.

Inlining		HTTP/1.1			HTTP/2.0					
		raw	active		raw		active		Server push	
		PLT [ms]	PLT [ms]	gain [%]	PLT [ms]	Gain [%]	PLT [ms]	Gain [%]	PLT [ms]	Gain [%]
DSL	Chrome PC	148.0	149.0	−0.68%	139.0	6.08%	147.0	0.68%	134.0	9.46%
	Firefox	238.5	173.0	27.46%	171.5	28.09%	154.5	35.22%	175.0	26.62%
	Chrome mobile	287.5	246.5	14.26%	272.0	5.39%	244.0	15.13%	268.0	6.78%
3G	Chrome PC	449.0	426.0	5.12%	645.0	−43.65%	428.5	4.57%	624.0	−38.98%
	Firefox	1889.0	1233.0	34.73%	1497.0	20.75%	1427.5	24.43%	1444.0	23.56%
	Chrome mobile	850.0	509.5	40.06%	850.0	0.00%	508.5	40.18%	683.0	19.65%

If 3G users make up a significant share of the website, it is highly recommended to adhere to the technique (the highest PLT gain for the "active" version, the biggest difference visible for Chrome PC: −32.65% gain for the "raw" version, 4.57% for the "active" version, −38.98% for the "server push" version).

With increased participation of users with permanent Internet access and using Google Chrome browser, consider switching to the server push technique (increase by 8.78 pp in relation to the "active" version and 3.38 pp in relation to the "raw" version).

Domain Sharding. The domain sharding technique only worked in the 3G user scenario, equipped with the Google Chrome browser (40.07% PLT gain for the "active" version, 14.34% gain for the "raw" version) (Table 3).

Table 3. Comparison of Page Load Time for **sharding** scenarios.

Domain sharding		HTTP/1.1			HTTP/2.0			
		raw	active		raw		active	
		PLT [ms]	PLT [ms]	Gain [%]	PLT [ms]	Gain [%]	PLT [ms]	Gain [%]
DSL	Chrome PC	251.0	257.0	−2.39%	215.0	14.34%	255.0	−1.59%
	Firefox	345.5	406.5	−17.66%	347.5	−0.58%	505.0	−46.16%
3G	Chrome PC	1078.0	478.0	55.66%	1047.5	2.83%	646.0	40.07%
	Firefox	2118.0	1830.0	13.60%	1703.0	19.59%	2256.0	−6.52%

If user participation is largely composed of this type of profile, applying the technique may be considered. Otherwise, it is recommended to abandon the

technique for the use of a single domain (gain decreases of up to 45.58 pp in relation to the "raw" version).

Minification. The situation with the use of minification is clear (Table 4). It is strongly recommended to continue using the technique to reduce the size of resources, which in each scenario yields a positive result in the context of PLT (gain increase of 79.52 pp compared to the "raw" version for the DSL profile and 143.97 pp in relation to the "raw" version for the 3G profile).

Table 4. Comparison of Page Load Time for **minification** scenarios.

Minification		HTTP/1.1			HTTP/2.0			
		raw	active		raw		active	
		PLT [ms]	PLT [ms]	Gain [%]	PLT [ms]	Gain [%]	PLT [ms]	Gain [%]
DSL	Chrome PC	7181.5	1727.5	75.95%	7464.0	−3.93%	1753.0	75.59%
	Firefox	7211.5	1766.5	75.50%	7531.0	−4.43%	1876.0	73.99%
	Chrome mobile	7581.0	2073.0	72.66%	7866.0	−3.76%	2014.5	73.43%
3G	Chrome PC	8411.0	1518.5	81.95%	15915.5	−89.22%	3806.0	54.75%
	Firefox	11121.0	3337.0	69.99%	17922.0	−61.15%	4970.0	55.31%
	Chrome mobile	9246.0	1902.0	79.43%	16496.0	−78.41%	4074.5	55.93%

5 Summary and Directions of Further Work

Properly designed, isolated simulation environment allowed to perform precise performance measurements of the websites examined in terms of the speed of loading the site during the first visit. Based on a precise analysis of the cleaned measurement data, it was possible to draw conclusion concerning the benefits of using classic techniques for designing efficient websites, including the HTTP/2 protocol. Based on these conclusions, a number of proposals were formulated regarding changes in the approach to already designed sites in terms of efficient charging using the old version of the protocol.

It turns out that not all techniques used in the context of the HTTP/1.1 protocol are beneficial after switching to the new version of the protocol. However, the recipients of the site should also be taken into account. Depending on the browser used or the available internet connection, the performance result may be different, as shown by many examples of specific visit scenarios.

For the concatenation technique, the only scenario convincing one to stick to it was the DSL scenario using the Firefox browser. In all other cases, the results were worse compared to the version without using the technique.

In the resource inlining scenarios using the 3G network proved to be beneficial. In the case of DSL networks, the use of the Google Chrome browser in the desktop version showed better results when using the alternative technique for resource inlining, i.e. server push.

The domain sharding technique proved to be useful only for a 3G user profile using the Google Chrome browser. Any other scenario indicated resignation from the technique as a beneficial step to increase efficiency.

The minification confirmed its "evergreen solution" title. The use of the technique did not lose its meaning after changing the protocol version to HTTP/2.

To sum up, when planning changes to the site design motivated by the use of the new version of the HTTP protocol, it is necessary to consider the nature of the site and who its recipient is, as it can have a significant impact on the final decision.

In the course of designing the simulation environment, the adoption of findings regarding the method of measuring size, the characteristics of visits, the interpretation of data, a number of aspects were found that can be subjected to improvements or changes. After their application, it shall be possible to conduct even more valuable substantive research and to draw new conclusions from them.

One of the aspects is to move the server environment to the cloud to examine real conditions. It would be necessary to conduct a larger number of measurements in the longer term, due to the fluctuations associated with the real network environment.

Another aspect is to examine a greater number of classic techniques that reveal their properties and benefits only in real network conditions.

The final suggestion for further work would be to further broaden the scenarios with new client profiles (browsers, devices) as well as network profiles (for example 4G, LTE and the like).

References

1. Grigorik, I.: High Performance Browser Networking. What Every Web Developer Should Know About Networking and Web Performance. O'Reilly Media, Sebastopol (2013)
2. Grigorik, I.: HTTP/2: A New Excerpt from High Performance Browser Networking. O'Reilly Media, Sebastopol (2015)
3. Liu, Y., Liu, X., Huang, G.: Can HTTP/2 really help web performance on smartphones? In: 2016 IEEE International Conference on Services Computing (2016)
4. Corbel, R., Stephan, E., Omnes, N.: HTTP/1.1 pipelining vs HTTP2 in-the-clear: performance comparison. In: 3th International Conference on New Technologies for Distributed Systems (2016)
5. de Saxcé, H., Oprescu, I., Chen, Y.: Is HTTP/2 Really Faster Than HTTP/1.1? In: 18th IEEE Global Internet Symposium (2015)
6. Varvello, M., Schomp, K., Naylor, D., Blackburn, J., Finamore, A., Papagiannaki, K.: Is the Web HTTP/2 Yet? In: 17th International Conference Of Passive and Active Measurement, PAM 2016 (2016)
7. Zarifis, K., Holland, M., Jain, M., Katz-Bassett, E., Govindan, R.: Modeling HTTP/2 speed from HTTP/1 traces. In: 17th International Conferenceof Passive and Active Measurement, PAM 2016 (2016)

8. 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: Proceedings of 37th International Conference on Information Systems Architecture and Technology – ISAT 2016 – Part I. AISC, vol. 521, pp. 235–247. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-46583-8_19
9. Rosen, S., Han, B., Hao, S., Morley Mao, Z., Qian, F.: Push or request: an investigation of HTTP/2 server push for improving mobile performance. In: The 26th International Conference (2017)
10. Kim, H., Lee, J., Park, I., Kim, H., Yi, D., Hur, T.: The upcoming new standard HTTP/2 and its impact on multi-domain websites. In: Asia-Pacific Network Operations and Management Symposium (2015)
11. Seemakhupt, K., Piromsopa, K.: When should we use HTTP2 multiplexed stream? In: 13th International Joint Conference on Computer Science and Software Engineering (2016)
12. Prokopiuk, J., Nowak, Z.: The influence of HTTP/2 on user-perceived Web application performance. Studia informatica, vol. 38, no. 3(132). Silesian University of Technology Press, Gliwice (2017)
13. Souders, S.: High Performance Web Sites. Essential Knowledge for Front-End Engineers. O'Reilly Media, Sebastopol (2007)
14. Ndegwa, A.: What is Page Load Time? MaxCDN One (2016). https://www.maxcdn.com/one/visual-glossary/page-load-time/
15. Dutton, S.: Measuring page load speed with navigation timing, blog HTML5 rocks (2011). https://www.html5rocks.com/en/tutorials/webperformance/basics/
16. Grigorik, I.: Measuring network performance with resource timing API, blog Google developers (2013). https://developers.googleblog.com/2013/12/measuring-network-performance-with.html
17. Trosien, O.: HTTP2 w/ Spring Boot+Jetty. GitHub (2017). https://github.com/otrosien/demo-http2
18. Rizzo, L.: The Dummynet Project. University of Pisa (2015). http://info.iet.unipi.it/~luigi/dummynet/

Teleinformatics and Telecommunications

Modified OMP Algorithm for Compressible Channel Impulse Response Estimation

Grzegorz Dziwoki(✉), Marcin Kucharczyk, and Jacek Izydorczyk

Institute of Electronics, Silesian University of Technology, Gliwice, Poland
{grzegorz.dziwoki,marcin.kucharczyk,jacek.izydorczyk}@polsl.pl

Abstract. Wireless communication link at the physical layer can be described as a signal transmission over multiple propagation paths which are different in the gain and the delay. But in reality there are only a few significant paths responsible for the most signal energy transmission between transmitter and receiver. Such a sparse nature of the propagation environment promotes the use of the compressed sensing methods for channel impulse response estimation in the receiver. Unfortunately, a typical band-limited transmission violates the strict channel sparsity making the impulse response reconstruction more complex. The paper presents an analysis of channel impulse response estimation with the Orthogonal Matching Pursuit algorithm. A modification of the classical OMP method is proposed in order to improve both the channel estimation and the data transmission qualities in case of a weak condition of the impulse response sparsity. This proposition is numerically evaluated for the general case of the OFDM transmission over a sixth path urban channel model. The results are compared to the ones obtained for the strict sparse impulse response instance.

Keywords: Channel estimation · Compressible impulse response
Compressive Sensing · Greedy methods · Orthogonal Matching Pursuit

1 Introduction

Wireless propagation channel consists of paths that carry signals from transmitter to receiver. Many experimental measurements of the wireless channels show that the vast majority of signal energy is delivered to the receiver only over a few isolated propagation paths, which are usually distant from each other in time [1,2]. This phenomenon can manifest afterwards in the recovered Channel Impulse Response (CIR) as a sparsity feature. The wider spectrum bandwidth of the transmitted signals, the more sparse pattern of the channel impulse response can be detected in the receiver. In case of the strict sparsity (a frequent assumption in many simulation analysis [3–6]) only several CIR taps are considered nonzero, making the Compressive Sensing (CS) processing effective in their estimation. For example, the greedy methods, such as Orthogonal Matching

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 161–170, 2018.
https://doi.org/10.1007/978-3-319-92459-5_13

Pursuit (OMP) [7] and Compressive Sampling Matching Pursuit (CoSaMP) [8] (both methods with their many modifications), are the popular CS algorithms that implementation is analyzed in many applications [9]. They work iteratively, making selection of a single impulse response element (or more in case of the CoSaMP) in each iteration step. Next, within the same iteration, the selected impulse response time index is merged with the ones found previously and all the taps amplitude that correspond to them are estimated by the Least Square (LS) algorithm simultaneously.

The above assumption about the strict CIR sparsity is quite uncertain in reality. A band-limited transmission due to the use of electronic filters in the transceivers, and some synchronization issues disperse the transmitted energy across the almost all taps of the impulse response. Consequently, many medium- and low-amplitude elements beside a few main ones appear in CIR. Such a signal structure, which represents the impulse response, is called *compressible* according to the compressed sensing terminology [10].

The channel compressibility reduces efficiency of the classic CS methods. The positions of nonzero elements, especially those with low amplitude can be incorrectly determined due to the noise presence and the insufficient number of measurement samples. Additionally, including a poor effect of the noise averaging on the CIR amplitude estimation, a final quality of the channel reconstruction may decrease.

The paper proposes a modification of the OMP method in case of compressible CIR reconstruction. The COST-207 TU6 channel model [1] was picked as a test environment in the simulation analysis. Channel estimation performance was evaluated in terms of Minimum Square Error (MSE), whereas transmission quality as Bit Error Rate (BER). The results obtained by the proposed solution were compared with the ones by other methods based on the OMP. Efficiency of the proposed modification in case of strict impulse response sparsity was tested as well.

The reminder of this paper is organized as follows. A comparison between sparse and compressible channel impulse responses with an explanation of the source of the compressibility is presented in Sect. 2. Section 3 depicts the CS processing with OMP method in relation to channel identification problem with a description of the proposed modification. Section 4 presents a simulation environment and the performance analysis of the proposed solution.

2 Sparse Channel vs Compressible Channel

Transmitter and receiver filters, including respectively, ones, transform a sparse propagation environment into a continuous-time channel impulse response. The resultant CIR of the transmission link can be described in baseband as:

$$h(t) = \sum_{l=1}^{L} a_l g(t - \tau_l) \tag{1}$$

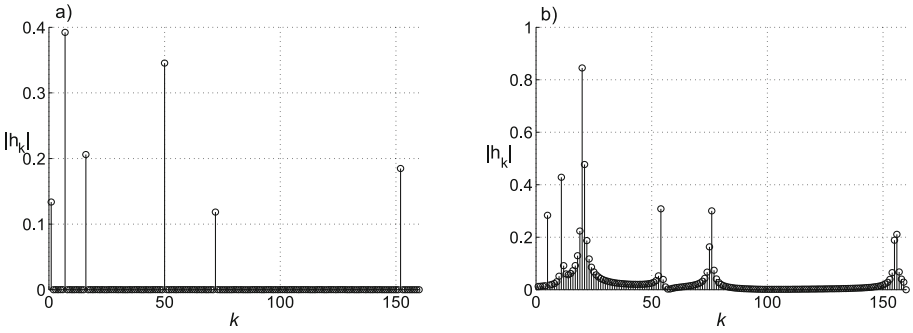


Fig. 1. Sparse channel impulse response (a) vs. compressible channel impulse response (b)

where a complex function $g(t)$ represents the joint impulse response of the transmitter and receiver linear processing blocks and a_l denotes complex gain of the lth propagation path that is delayed by time τ_l. L is the total number of the resolvable paths. Because the receiver samples incoming signal with period T_S, it is more convenient for processing purposes to present the CIR in discrete-time domain ($h_k = h(kT_S)$) using a vector notation:

$$\mathbf{h} = [h_0, h_1, \ldots, h_k, \ldots, h_{K-1}]^T \tag{2}$$

where k is a discrete-time index and K corresponds to the maximum delay spread $\Delta_{max} = KT_S$ of the channel impulse response.

If all the path delays τ_l are consistent with the sampling period i.e. $\tau_l = kT_S$ and $g(t)$ is a Nyquist filter, then $\mathbf{h}$ can be kept exactly as sparse as the propagation environment provided that appropriate time synchronization is maintained. But real-life situations are more complex and there are usually much more non-zero elements (often even all of them) in $\mathbf{h}$ due to a cumulative influence of the synchronization issues and the path delays combination. Consequently, the high amplitude elements of $\mathbf{h}$ form clusters at the positions related to delays which are in the close vicinity to the delays attributed to the actual propagation paths. On the other hand, due to time domain shape of $g(t)$, the amplitudes within the clusters tend to decrease as the time span from the actual propagation delays increases. An example of compressible CIR for a sparse propagation environment is depicted in Fig. 1b. The sparse counterpart is presented in Fig. 1a.

3 CIR Identification Using the OMP Nethod

3.1 The Classic Approach

Consider a discrete-time baseband transmission system that is described with matrix notation. A $M \times 1$ complex measurement vector $\mathbf{z} \in \mathbb{C}^M$ contains signal

samples of the transmission channel output collected within a single acquisition period (e.g. a training period). It is linearly related to the channel impulse response $\mathbf{h} \in \mathbb{C}^K$ via convolution operation:

$$\mathbf{z} = \mathcal{X}\mathbf{h} + \mathbf{v}, \tag{3}$$

where $\mathcal{X}$ is a $M \times K$ complex Toeplitz measurement matrix and $\mathbf{v}$ denotes a complex vector with samples of the system noise. The elements $x_{m,k}$ of the measurement matrix $\mathcal{X}$ come from a randomly generated training signal with good autocorrelation property. The training signal is known at the receiver.

According to the sparsity property, the channel impulse response $\mathbf{h}$ is an unknown S-sparse vector i.e. no more than S of its components have nonzero values. The OMP method (like any other CS algorithm) uses sparsity to detect nonzero path amplitudes in the channel impulse response. It is essential, because if $M < K$ than the system of linear equations:

$$\mathbf{z} = \mathcal{X}\hat{\mathbf{h}}, \tag{4}$$

is underdetermined, so the estimator $\hat{\mathbf{h}}$ of the channel impulse response cannot be directly determined using the LS estimation method [11]. A solution is to go through successive iterations of the CS procedure. In a single iteration loop, the OMP algorithm finds approximation of one of the current path delays using following maximization:

$$l = \underset{l \in \{0,1\ldots,K-1\}}{\arg\max} \; {\varepsilon^{(j)}}^H \mathcal{X}. \tag{5}$$

where j is the iteration number and $\varepsilon^{(j)}$ is the residual error in the j-th iteration that is updated according to the formula:

$$\varepsilon^{(j)} = \mathbf{z} - \mathcal{X}\hat{\mathbf{h}}^{(j)}. \tag{6}$$

In every iteration loop the effective size of $\mathcal{X}$ is only $M \times j$ with $M > j$ because $K - j$ elements in the $\hat{\mathbf{h}}^{(j)}$ are zero. In that case the Moore-Penrose pseudoinverse of $\mathcal{X}_{(M\times j)}$ can be determined and the LS estimation of channel impulse response is:

$$\hat{\mathbf{h}}^{(j)}_{(j\times 1)} = (\mathcal{X}^H_{(M\times j)}\mathcal{X}_{(M\times j)})^{-1}\mathcal{X}^H_{(M\times j)}\mathbf{z} \tag{7}$$

where $\hat{\mathbf{h}}^{(j)}_{(j\times 1)}$ is a concise representation of $\hat{\mathbf{h}}^{(j)}$, i.e. only nonzero values of the estimated channel impulse response are represented in the vector.

The OMP procedure stops after predefined number of iterations (if the sparsity S is known) or when a stopping condition is met. The value of $\hat{\mathbf{h}}$ in the last iteration is the valid estimate of the channel impulse response, which can be used for further signal processing in the receiver. Detailed description of the OMP processing steps can be found in [7].

Compressed sensing theory establishes a suggested minimum amount M of measurement samples $\mathbf{z}$ that are required if high probability of sparse signal estimation is expected. This value is proportionally related to the following expression:

$$M \propto S \ln K. \tag{8}$$

If M increases then the higher precision of the paths delay approximation and noise suppression are obtained reducing the MSE error, which is defined as follows:

$$\text{MSE} = ||\hat{\mathbf{h}} - \mathbf{h}||_2^2. \tag{9}$$

3.2 A Modification for Compressible CIR

The noise $\mathbf{v}$ enhances ambiguity of the channel reconstruction by the OMP. Among all the impulse response taps, those of them with low amplitudes, in particular, are the most prone to estimation errors. In many cases the OMP algorithm can not assign their correct time index positions due to a masking effect of the channel noise. Consequently, contrary to the expectations, the current MSE value can rather grow than go down. To minimize the probability of the incorrect CIR estimation, a two-threshold stopping criterion was proposed in [12,13]. It measures energies of the residual error ε and of the difference between successive ones, and compares them to a predefined thresholds. When the residual error ε does not change significantly through the consecutive iterations due to incorrect tap identification, the OMP procedure can be interrupted. This stopping moment can be described mathematically as follows:

$$\text{STOP} \quad if \quad \begin{cases} j > S_{max}, & \text{any } \| \varepsilon^{(j)} \|_2^2, \| \varepsilon^{(j)} - \varepsilon^{(j-1)} \|_2^2 \\ \| \varepsilon^{(j)} \|_2^2 < aM\sigma^2, & \text{any } \| \varepsilon^{(j)} - \varepsilon^{(j-1)} \|_2^2 \\ \| \varepsilon^{(j)} - \varepsilon^{(j-1)} \|_2^2 < b\sigma^2, & aM\sigma^2 \leq \| \varepsilon^{(j)} \|_2^2 < caM\sigma^2 \end{cases} \tag{10}$$

where a, b and c are some scaling factors which should be chosen according to specific conditions of the recovered channel, j is the current iteration step and σ^2 is noise power.

The above-mentioned control mechanism works fine for sparse channels (signals) [13]. But in case of compressible ones, there is a danger that the iterative procedure can stop too early (see a discussion of the simulation results for details in the next section) before some crucial taps with medium and low amplitude are recovered. They are necessary for preserving transmission quality. Their positions are clearly indicated by the channel compressibility pattern which is related to $g(t)$. Unfortunately, the classic OMP procedure does not use this knowledge to control how to select the time index of the next element in the recovered channel impulse response.

The aim of the paper is to propose a solution that improves quality of CIR reconstruction by using the compressibility pattern as an additional guideline to control the OMP algorithm. No information about the shape of $g(t)$ is provided. The proposed modification combines the two-threshold stopping criterion

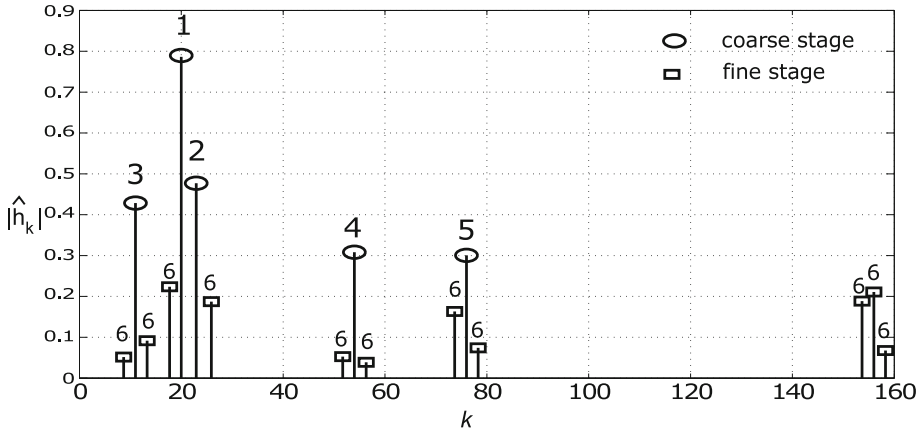


Fig. 2. Step-by-step reconstruction by the modified OMP with $l = 1$

approach with a block CS processing idea [14]. Two phases can be distinguished in it. Initially after the start of the estimation procedure, the consecutive CIR elements are recovered according to the classic OMP. This coarse recovery stage is valid until the energy of the residual error falls below the predefined threshold. When the condition $||\varepsilon^{(j)}||_2^2 < caM\sigma^2$ is met, the fine stage of reconstruction begins. From that moment, new elements of the reconstructed impulse response are selected in each next iteration step using an additional selection rule besides the classical one. The OMP algorithm stops ultimately when $||\varepsilon^{(j)}||_2^2 < aM\sigma^2$ or $||\varepsilon^{(j)} - \varepsilon^{(j-1)}||_2^2 < b\sigma^2$.

The detailed description of the iteration step in the fine stage is as follows:

- find a new time index of the CIR element according to Eq. (5);
- supplement the set of the time indices that have been already found in the current and the previous iteration steps with subsequent ones according to a nearest neighbor principle. This rule complies with the compressibility pattern so new elements of the recovered CIR surrounds the old ones within the distance range of l taps;
- update amplitude of all the selected CIR elements according to the LS method;

Figure 2 presents graphically the step-by-step reconstruction of a channel impulse response with proposed modification of the OMP method and $l = 1$. The bars marked with circle on the top are the impulse response elements recovered in the coarse stage. The remaining ones are added in the fine stage. The numbers on the top of each bar explain the order of the CIR taps recovery (the successive iteration numbers).

4 Numerical Analysis

4.1 Simulation Environment

Consider a hypothetical Cyclic Prefixed Orthogonal Frequency Division Multiplex (CP-OFDM) system with symbol duration T_B and sampling period T_S. The P consecutive OFDM symbols form a single transmission frame that is transmitted through a linear multipath channel described by the impulse response $\mathbf{h}$. The channel is considered static during transmission of a single frame.

The frame is preceded by a pseudorandom training preamble that is used for synchronization and channel identification purposes. The length of the preamble is at least two times longer than maximum delay spread of channel impulse response. This condition ensures that the second half of the training sequence is free of influence of the preceding frame and all the received samples within this part can be used in the compressed sensing identification procedure. According to the notation in Sect. 3.1, these are the measurement samples described by the vector $\mathbf{z}$. The matrix representation of the training sequence is denoted as $\mathcal{X}$.

Details of the remaining major assumptions of the simulation system model are as follows:

- The matrix $\mathcal{X}$ is square with size $M \times M$ where $M = 160$. It corresponds to the guard period size in the LTE transmission. The matrix elements are pseudorandom samples with power $\sigma_x^2 = 1$;
- The COST 207 model of multipath wireless channel for terrestrial propagation in an urban area is considered [1]. It consists of six active propagation path. The detailed model description is presented in Table 1. The target channel impulse response is shaped by $\sin(x)/x$ Nyquist filter with double-sided bandwidth of $f_S = 1/T_S$ in order to simulate the band-limited transmission;
- signal-to-noise ratio SNR $= \sigma_X^2/\sigma_v^2$ varies in the range of 5[dB] to 25[dB] with 2[dB] step;
- Any subcarrier in the OFDM symbol is not used as the pilot and guard tone. The 4-QAM constellation is used for data coding in the subchannels;
- the quality metrics, MSE and BER, are evaluated as an average of the transmissions over 50 randomly generated channels which are comply with the assumed model;

Table 1. TU6 profile

Tap number	Delay [μs]	Power [dB]
1	0	−3
2	0.2	0
3	0.5	−2
4	1.6	−6
5	2.3	−8
6	5	−10

– the scaling factors in the stopping criterion (10) are $a = 1$, $b = 10$ and $c = 2$ according to the results in [13];
– the vicinity range $l = 1$;

4.2 Results and Discussion

The goal of the performed simulations was to test the compressed sensing methods (along with the proposed modification) in compressible environment represented by the channel impulse response. The same processing core, which was originally designed for sparse signals reconstruction, was the common factor of the four OMP versions selected and compared to each other during the simulations. These methods are labeled as follows:

– **OMP** - the classic OMP method with the stopping criterion defined by the energy of the residual error only;
– **s-OMP** - the OMP method with the stopping criterion described by Eq. (10);
– **b-OMP** - the block OMP with the stopping criterion described by Eq. (10)
– **sb-OMP** - the proposed two-phase OMP - the combination of **s-OMP** and **b-OMP** methods;

Among all the analyzed solutions, the proposed one achieves the best results in terms of both the MSE and the BER metrics in wide range of the considered SNR as long as the compressible channel is recovered. For example, assuming BER = 0.01 (SNR about 20 [dB]), it is about 20% of the transmitted energy that can be saved if **sb-OMP** is used in comparison to **s-OMP** and about 8% in relation to the remaining methods (see Fig. 4). Moreover the MSE of **sb-OMP**, in particular in the medium range of SNR, is located in close vicinity of the minimum MSE values(**MMSE**). This minimum estimation reference level was evaluated numerically for the best set of CIR elements with their amplitudes computed by LS algorithm. The full knowledge of the channel characteristic was necessary to determine this theoretical limit.

Because the proposed modification had been developed for use in case of compressible channels (the common situation occurring during signal transmission) it was to be expected that for the strict sparse CIR instances, its quality results may be worse as it is depicted in Fig. 3. The possible absence of the impulse response elements just next to the main ones in the sparse channel is the primary reason of the estimation performance reduction when block reconstruction strategy is used in OMP. Fortunately, this situation is not a big problem in practice. The probability that a current channel instance has sparse impulse response is rather very low.

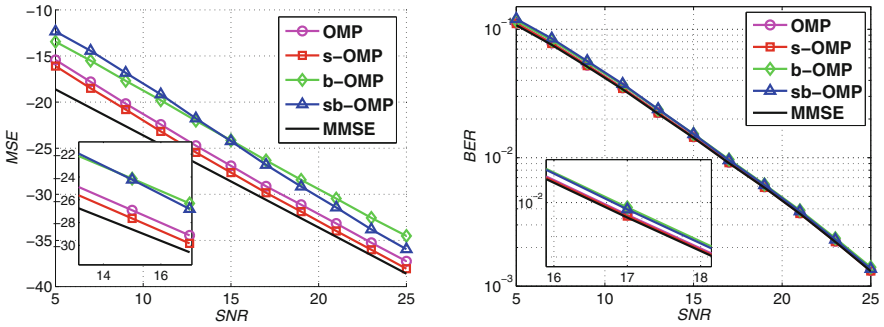


Fig. 3. Performance comparison for strict sparse CIR estimation

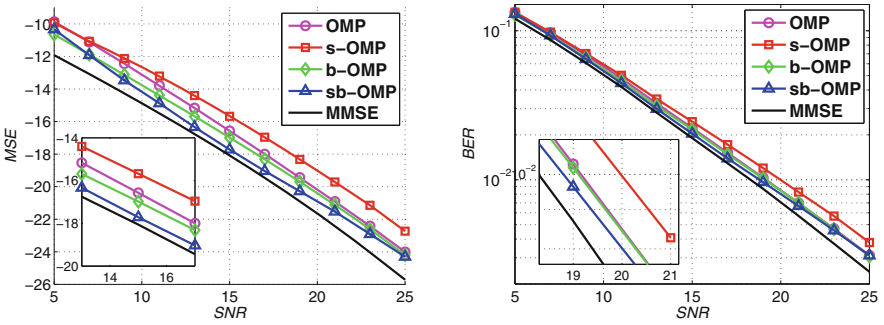


Fig. 4. Performance comparison for compressible CIR estimation

5 Conclusions

The paper presents analysis of the compressible channel impulse response reconstruction using the compressed sensing estimation algorithms. The modifications of the OMP method, including the new one, were investigated and compared in the numerical experiments. The proposed solution combines the classic OMP processing in the early stage of the reconstruction procedure with the grouped one when the residual error energy is sufficiently low, i.e. when the most significant channel taps have been already recovered. The obtained results show the performance improvement in terms of the MSE and the BER metrics.

The analysis was performed assuming no knowledge about the time-domain shaping pattern introduced by the band-limited transmission. A direct identification of the propagation paths instead of the samples of the CIR is planned as a future work.

Acknowledgement. This work was supported by the Ministry of Science and Higher Education funding for statutory activities (BK-232/RAu-3/2017).

References

1. Failli, M.: Digital land mobile radio communications COST 207. Technical report, European Commission (1989)
2. Guidelines for evaluation of radio transmission technologies for imt-2000. Technical Report Rec. ITU-R M.1225, ITU (1997)
3. Dai, L., Wang, J., Wang, Z., Tsiaflakis, P., Moonen, M.: Spectrum- and energy-efficient OFDM based on simultaneous multi-channel reconstruction. IEEE Trans. Signal Process. **61**(23), 6047–6059 (2013). https://doi.org/10.1109/TSP.2013.2282920
4. Dai, L., Wang, Z., Yang, Z.: Compressive sensing based time domain synchronous OFDM transmission for vehicular communications. IEEE J. Sel. Areas Commun. **31**(9), 460–469 (2013). https://doi.org/10.1109/JSAC.2013.SUP.0513041
5. Ding, W., Yang, F., Pan, C., Dai, L., Song, J.: Compressive sensing based channel estimation for OFDM systems under long delay channels. IEEE Trans. Broadcast. **60**(2), 313–321 (2014). https://doi.org/10.1109/TBC.2014.2315913
6. Dziwoki, G., Izydorczyk, J.: Iterative identification of sparse mobile channels for TDS-OFDM systems. IEEE Trans. Broadcast. **62**(2), 384–397 (2016). https://doi.org/10.1109/TBC.2016.2529288
7. Tropp, J., Gilbert, A.: Signal recovery from random measurements via orthogonal matching pursuit. IEEE Trans. Inf. Theor. **53**(12), 4655–4666 (2007). https://doi.org/10.1109/TIT.2007.909108
8. Needell, D., Tropp, J.: Cosamp: iterative signal recovery from incomplete and inaccurate samples. Appl. Comput. Harmonic Anal. **26**(3), 301–321 (2009). https://doi.org/10.1016/j.acha.2008.07.002
9. Qaisar, S., Bilal, R.M., Iqbal, W., Naureen, M., Lee, S.: Compressive sensing: from theory to applications, a survey. J. Commun. Netw. **15**(5), 443–456 (2013). https://doi.org/10.1109/JCN.2013.000083
10. Candes, E., Wakin, M.: An introduction to compressive sampling. IEEE Signal Process. Mag. **25**(2), 21–30 (2008). https://doi.org/10.1109/MSP.2007.914731
11. Haykin, S.: Adaptive Filter Theory, 5th edn. Pearson, Harlow (2014)
12. Dziwoki, G., Izydorczyk, J.: Stopping criteria analysis of the OMP algorithm for sparse channels estimation. In: Gaj, P., Kwiecień, A., Stera, P. (eds.) CN 2015. CCIS, vol. 522, pp. 250–259. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19419-6_24
13. Dziwoki, G.: Averaged properties of the residual error in sparse signal reconstruction. IEEE Signal Process. Lett. **23**(9), 1170–1173 (2016). https://doi.org/10.1109/LSP.2016.2588728
14. Baraniuk, R.G., Cevher, V., Duarte, M.F., Hegde, C.: Model-based compressive sensing. IEEE Trans. Inf. Theor. **56**(4), 1982–2001 (2010). https://doi.org/10.1109/TIT.2010.2040894

Evaluation of the Jammers Performance in the WiFi Band

Dariusz Czerwinski[1(✉)], Jaroslaw Nowak[2], and Slawomir Przylucki[1]

[1] Lublin University of Technology, 38A Nadbystrzycka Str., 20-618 Lublin, Poland
{d.czerwinski,s.przylucki}@pollub.pl
[2] Polish Air Force Academy, Dywizjonu 303 no. 35 Str., 08-521 Deblin, Poland
j.nowak@wsosp.pl

Abstract. The paper presents evaluation of the performance of two different jammers in an open space on the basis of measurements. The measurements were performed with the use of Fluke AirMagnet Spectrum XT analyser in agricultural area. Signal power heat maps were prepared and analysed. Theoretical path loss models of different jammers' configurations were worked out. The determination of the path loss models parameters allow to calculate the theoretical heat maps and compare them with measurements. The study focused on designation of the performance of the jammers in open space. This allows to estimate the area of effective signal jamming. The influence of channel frequency in WiFi 2.4 GHz band on the jammer performance was analysed.

Keywords: Jammer performance · Path loss model
Signal power measurements

1 Introduction

Wireless networks are commonly used nowadays and cover the residential, industrial and corporate environments [1,2]. The 802.11 WiFi networks are an attractive target for various types of attacks. Recently scientists great interest is in attacks performed by jammers. There are a few types of these devices i.e.: random jammers, reactive jammers, constant jammers and smart jammers. Jamming attacks came to be an important issue today. It is essential to locate the position of jamming device to take security action and restore the communication [3–5]. Knowledge on jammer performance in various surrounding is a valuable information for anti-jamming purposes.

The effect of noise power jammer and follower jammer on the Bluetooth Personal Area Network was evaluated in [3]. Authors proved that performance of network depends on many factors such as: power density, distance or immunity technique. In another work the evaluation of the methodology for using the performance characterization as a tool for modelling the behaviour of 802.11p networks in the presence of RF jamming attacks was presented in [4].

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 171–182, 2018.
https://doi.org/10.1007/978-3-319-92459-5_14

Authors in [5] designed simple approach for locating the jammer through a set of measurable parameters. The key observation in this research was that the Packet Delivery Ratio (PDR) measured locally by a device decreases as a receiver moves closer to the jamming source. One of the used parameters was a path loss exponent. A study on the jammer location problem was conducted by Cheng et al. in [6]. The authors proposed a simple and effective algorithm called Double Circle Location (DCL). They made their research under different conditions including: various node densities, jammers' transmission powers and antenna orientations. The path loss propagation model with Gaussian noise was used for their calculations.

The Bayesian game with asymmetric information for interaction between the network and the smart jammer has been modelled with the use of path loss model in [7]. In the other research [8] the investigation of the jamming detection on smart-phones based on the received signal strength indicator (RSSI) and packet loss rate (PLR) was conducted. Recently it can be observed, that many solutions associated with locating of the device are based on the path loss models and processing of the values of RSSI indicator [9–11].

Our contribution, presented in this work consists of the measurements and evaluation of the performance of chosen jammers in WiFi band. Based on the measurement results, heat maps of signal power and theoretical path loss models (log-normal and two-ray ground reflection) were determined. On the basis of measurement results it can be seen that performance of jamming devices is different in different WiFi channels. The remainder of the paper is organised as follows: Sect. 2 describes the hardware configuration of the test stand for measurements of signal's power propagation as also scenarios. Section 3 describes the experiment results. Jammers' performance and path loss models are discussed in Sect. 4. Conclusions and some further research ideas are presented in Sect. 5.

2 Test Stand for Measurement of Signal's Power Propagation

Test stand for measurements of signal's power propagation of jammers consists of the following elements:

- CRJ4000 cell phone and WiFi handheld jammer [12],
- stationary jammer CKJ-1502A12 [13],
- AirMagnet Spectrum XT Fluke analyser for data measurements and RF analysis [14],
- Laptop PC with MS Windows 10 and AirMagnet WiFi Analyzer software.

The measurement test stand used in research is shown in Fig. 1. Measurements rely on recording data in WiFi 2.4 GHz frequencies with Fluke analyser, while the recorder changed position both in distance and placement angle which is shown in Fig. 2. Measurements presented in the article were performed in open area in the grassy terrain during autumn season. Results of jammer field propagation were presented for channels 1, 6 and 11 of WiFi 802.11b network.

Fig. 1. The measurement test stand

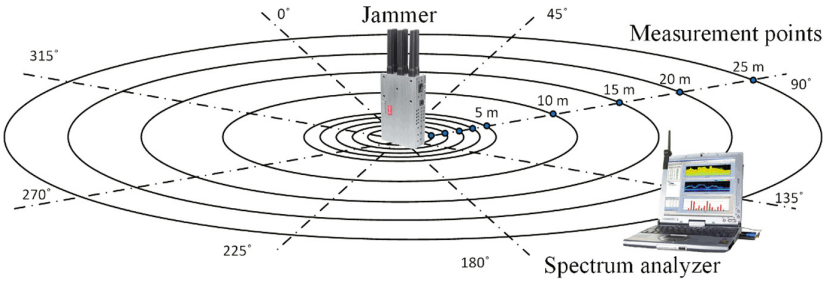


Fig. 2. Idea of measurements

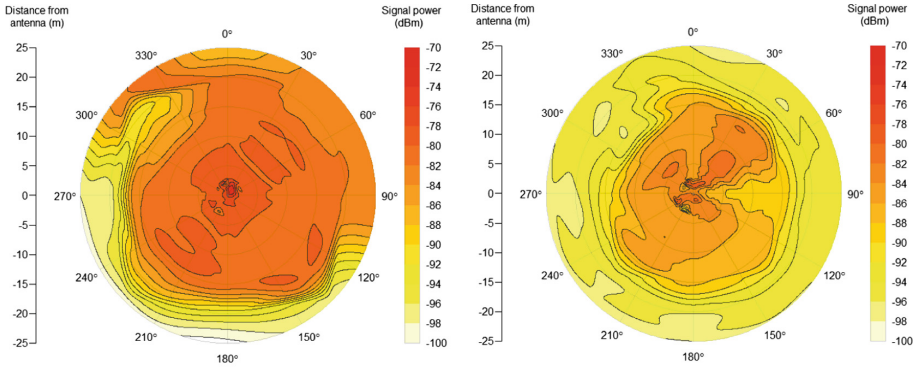
In order to perform measurements both jammers were configured to work in WiFi 2.4 GHz frequency ranges 802.11b standard. Because the CKJ-1502A12 jammer has an adjustable power it was configured in that way to produce the same power level as CRJ4000 at the length of 1m. The summary of parameters of both jammers were shown in the Table 1.

The Fluke (USB based) analyser has following parameters:

- freq. range: 2402–2494; 5160–5330; 5490–5710; 5735–5835; 4910–4990 MHz,
- operating temperature: 0 to 70 °C,
- DC power: voltage supply 5 V; active power: 2 W,
- amplitude accuracy: +/−2 dB,
- resolution Bandwidth: 156.3 KHz,
- sweep time: 64 msec per 20 MHz or 64 msec per channel.

Table 1. Summary of jammers parameters

Model	CRJ4000	CKJ-1502A12
Output power	2.5 W	up to 30 W
Shielding radius	up to 20 m at −75 dBm	2–40 m at −75 dBm
Frequencies	2400–2500 MHz	2400–2500 MHz
Antennas	Omnidirectional	Omnidirectional
Antennas gain	2.8 dBi (2.4 GHz)	9 dBi (2.4 GHz)
Operating temperature	15 °C to 60 °C	15 °C to 60 °C
Power supply	AC 110–240 V (50–60 Hz), DC 12 V, 1800 mAh Li-Ion	AC 110–240 V (50–60 Hz), DC 12 V



(a) (b)

Fig. 3. Measurement results for scenario 1: (a) channel 1, (b) channel 6

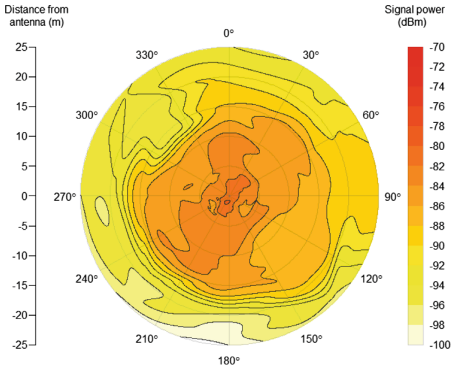


Fig. 4. Measurement results for scenario 1 and channel 11

The experiment relied on designation of the distribution of the signal emanated by the jammers in the open area in different configurations. Measurements were performed in the LOS (Line of Sight) conditions and prevailing temperature was in range 17–19 °C. For collecting the signal power values (in dBm) coming from jammer the analyser was placed at a distance of 1, 2, 3, 4, 5, 10, 15, 20 and 25 m. Both devices were placed on the wooden stands 62 cm above the ground. For each distance 8 omnidirectional measurements with analyser were performed at the angles of 0, 45, 90, 135, 180, 225, 270, and 315° as shown in Fig. 2. Measurements were performed in the WiFi 2.4 GHz wireless band. Before turning on the jammer the background noise was measured with the Fluke XT analyser. The value of this noise was in range −110 dBm to −105 dBm. At each measurement point the 100 measurements were made. Frequency range was from 2.402000 to 2.493875 GHz with 156 kHz step. The first measurements were carried out for jammer CKJ-1502A12 (scenario 1). The next scenario (2) used hand-held CRJ4000 jammer as a source of interference. The last, third scenario consists in taking measurements for both operating jammers placed close to each other, at the same wooden stand with 1 cm distance.

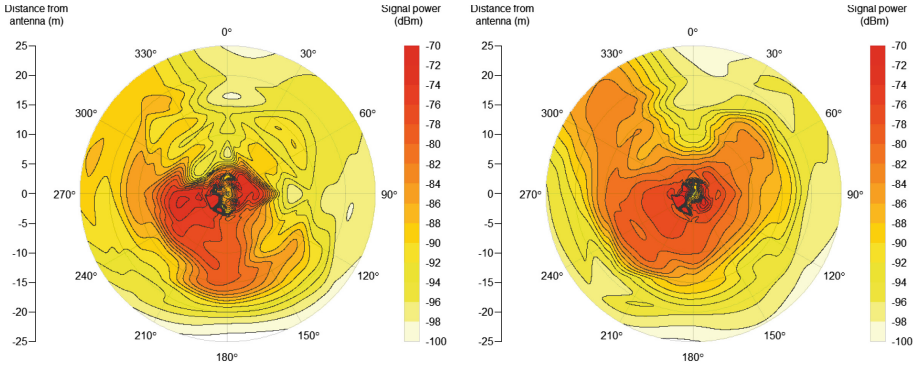


Fig. 5. Measurement results for scenario 2: (a) channel 1, (b) channel 6

3 Results of the Measurements

The average values were calculated for each measurement point and after this the interpolation was applied to obtain the heat polar maps. The maps were prepared in R environment with the use of the *akima* package.

Results of measurements for scenario 1 (CKJ-1502A12 stationary jammer) are presented in Figs. 3 and 4. The distribution of signal power generated by the jammer for channel 1 of 2.4 GHz frequency band is shown in Fig. 3a. For the other channels i.e. 6 and 11 the results are presented in Figs. 3b and 4 accordingly. It

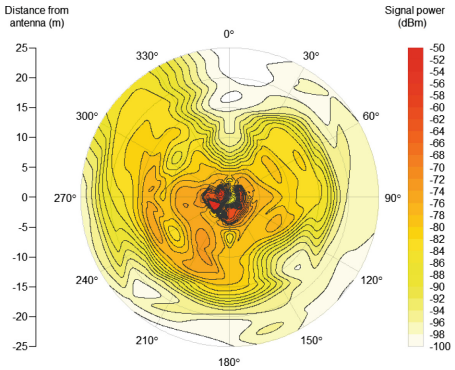


Fig. 6. Measurement results for scenario 2 and channel 11

can be noticed that much better area coverage occurs for channel 1 comparing to the other two channels. This may be caused by jammer hardware, which performs differently for WiFi frequencies.

Results of the measurements for hand-held jammer CRJ4000 are presented in Figs. 5 and 6. The distribution maps of the signal power for channels 1 and 6 are shown in Figs. 5a and b accordingly. Results for channel 11 are shown in Fig. 6. One can notice that for hand-held jammer best performance (highest values of signal power given in dBm) is obtained for channel 11 (values on right scale in Fig. 6 are higher).

Results of the measurements for both jammers working (scenario 3) are shown in Figs. 7 and 8. Figures 7a and b show the of the signal power maps for channels 1 and 6 accordingly. Results of the measurement for channel 11 are shown in Fig. 8. The values of signal power obtained from both jammers are higher (colour scales in Figs. 7 and 8) and distribution is more uniform comparing to scenarios 1 and 2. This was expected, due to the fact that two WiFi jamming sources will amplify the signal.

It can be noticed from Figs. 3, 4, 5, 6, 7 and 8 that distribution of the signal power around the jammers antennas is not uniform, despite the fact that used antennas were omnidirectional. Therefore in further modelling the authors decided to use average values calculated from measurements of different channels for each scenario.

4 Performance of the Jammers' and Their Path Loss Models

Path loss models for WiFi networks can be calculated in different ways. Some are obtained theoretically [6,15], but most of them are based on the measurements [4,16–18]. The presented research follows this methodology and presents models based on empirical research. The power of received signal in an open space is determined by the Friis's law and decreases with the square of distance [19,20]. In

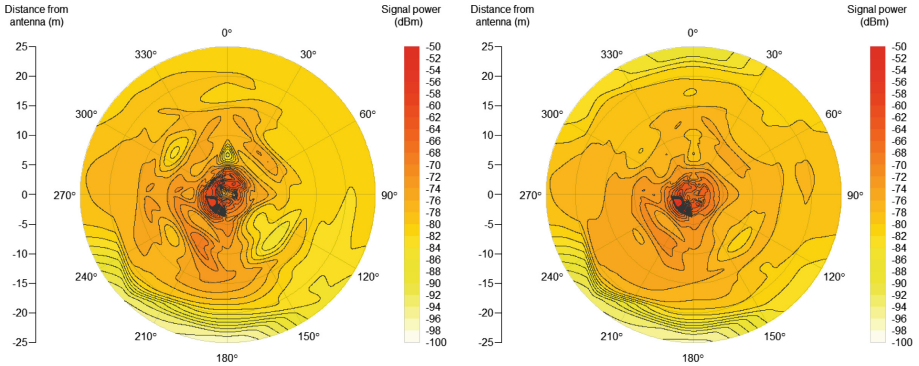


Fig. 7. Measurement results for scenario 3: (a) channel 1, (b) channel 6

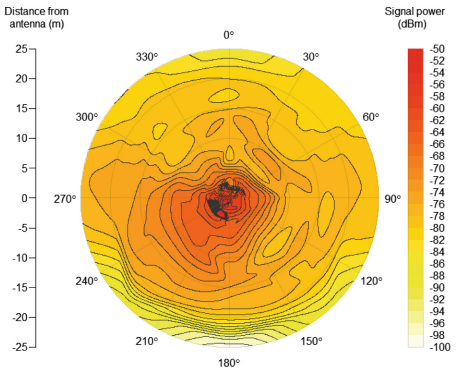


Fig. 8. Measurement results for scenario 3 channel 11

other environments the signal is attenuated in many different ways, i.e.: coming from obstacles between devices, reflections from the walls of buildings, ground reflections, other devices in the same band, etc. In such cases most commonly used path loss models are log-distance model with shadowing phenomenon and two-ray ground reflection model [21–24].

However if the transmitter and receiver in an open area are in the line of sight (LOS) and there are no obstacles between them, commonly used model is the classical log-distance one (1). This model introduces the attenuation factor, called a path loss exponent n, which is environmentally dependent [21–23].

$$P_r(d) = P_r(d_0) - 10 \cdot n \cdot log_{10}\left(\frac{d}{d_0}\right) \tag{1}$$

where: $P_r(d)$ is the signal power, $P_r(d_0)$ is the received signal power at the reference distance d_0, n path loss exponent, d is the distance from the transmitter.

In this paper additionally two-ray ground reflection model was taken into consideration. This model considers the impact of the radio wave signal reflected from the ground. The impact depends on the ground properties, height of antennas and Fresnel zone. Path loss in this model can be defined as expression (2) [4,21,24].

$$P_d(d) = P_r(d_0) - 10 \cdot log_{10} \left| \frac{1}{d} + \Gamma \cdot \frac{e^{j2\pi \frac{\delta_d}{\lambda}}}{d + \delta_d} \right| \tag{2}$$

where: $P_r(d)$ is the signal power, λ is a wave length, δ_d is additional reflected path given by Eq. (3) and Γ is coefficient of Fresnel reflection for the signal given by Eq. (4),

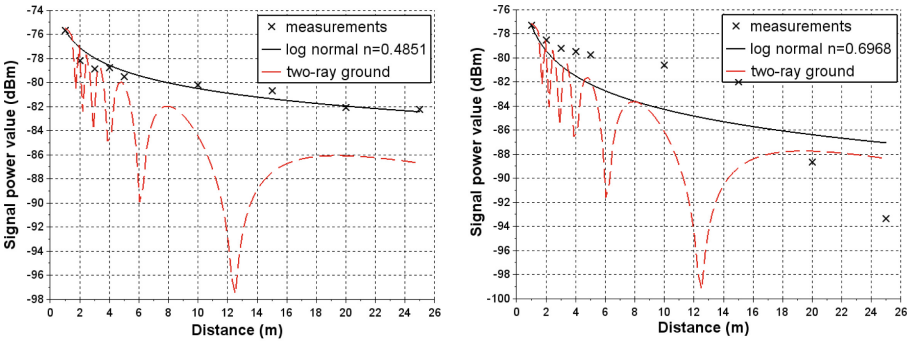
$$\delta_d = \sqrt{(h_t + h_r)^2 + d^2} - \sqrt{(h_t - h_r)^2 + d^2} \tag{3}$$

where: h_t, h_r are accordingly heights of antennas of the transmitter and receiver,

$$\Gamma = \frac{sin(\theta) - \sqrt{\varepsilon_r - cos(\theta)^2}}{sin(\theta) + \sqrt{\varepsilon_r - cos(\theta)^2}} \tag{4}$$

where: θ is the angle between the ground and reflected path, ε_r is ground relative permittivity.

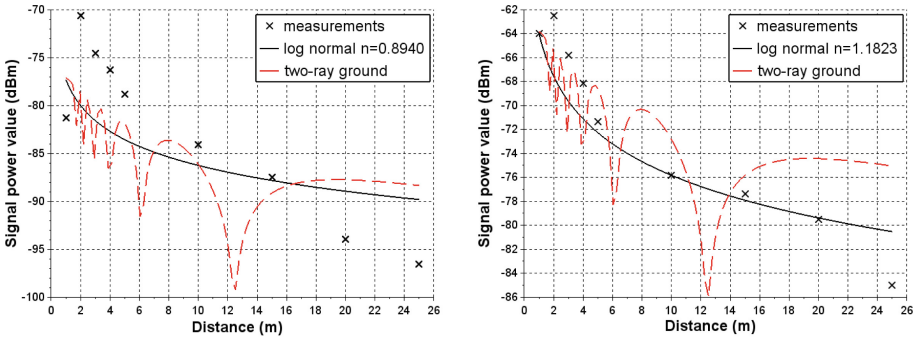
In further analysis the authors took into consideration two models corresponding to the open free space in the LOS conditions (Eqs. (1) and (2)). Based on the measurement results of the path loss exponent for different scenarios was calculated.



(a) (b)

Fig. 9. Signal power measurement results from scenario 1 and calculated path loss models versus distance: (a) channel 1, (b) average values for channels 1, 6, 11

Sample measurement results and calculated path loss models for scenario 1 and channel 1 were shown in Fig. 9a. The modelling results for average values for channels 1, 6, 11 are shown in Fig. 9b. The other results of log-normal shadowing model for scenario 2 and 3 are: $n = 0.8940$ for second scenario, $n = 1.1823$ for third scenario.

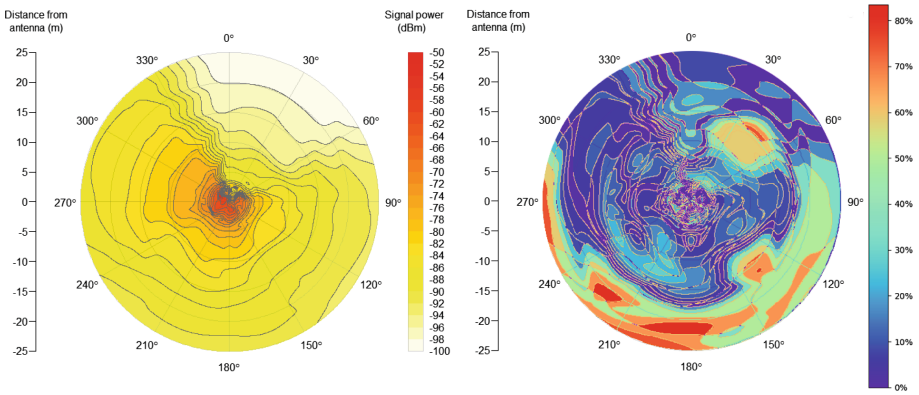


(a) (b)

Fig. 10. Average values of signal power measurement results and calculated path loss models versus distance: (a) scenario 2 - hand-held jammer, (b) scenario 3 - both jammers

Table 2. Summary of jammers performance distance

Scenario	Measurements distance (m)	Log-normal model distance (m)	Two-ray model distance (m)
1	<1	<1	<1
2	<3.5	<1	<1
3	<8.2	<8.8	<11.5 or >14



(a) (b)

Fig. 11. Comparison of heat maps values for scenario 2 channel 11: (a) values calculated from path loss models, (b) differences of heat maps measured and calculated given in %

Based on the measurements and simulation results (Figs. 9b and 10a, b) it can be noticed that value of path loss exponent n increases due to the faster signal damping. It can be seen that both models don't fit measurement data in whole range. Two-ray ground model better fits in the shorter ranges i.e. 1–5 m, while log-normal corresponds better to scenario 3 at distances grater than 5 m.

Measurement data and determined models allow to estimate the performance of the jamming devices in different scenarios. Parameters given by the manufacturers and summarised in Table 1 give an information about shielding radius when the band is jammed with signal of power equal to −75 dBm. In other research [4,25,26] it can be found that front-end saturation of the channel can be achieved with the jammer signal power of approximately −50 dBm. In this work authors decided to estimate the distance of successful jamming while the jammer signal power is in the range of $(-50--75)\,dBm$. The results of measurements and modelling were given in Table 2. It is clearly visible that shielding radius of the devices (Table 2, measurements results of scenarios 1 and 2) is smaller than given by manufacturers. In case of scenario 3 (both jammers working) shielding radius is in the range given in Table 1 due to the strengthening interference of devices. Thus authors decided to calculate signal power heat maps based on above assumption and compare them with heat maps obtained from measurements. For each measured angle both path loss models were evaluated and based on their results the heat maps were prepared. Two images were compared with Python software code and *numpy* package for scientific computing. As a sample of comparison the distribution of measured (Fig. 6) and calculated (Fig. 11a) signal power heat maps from scenario 2 was shown in Fig. 11b. It can be noticed that calculated values from path loss models do not differ much (less than 25%, Fig. 11b) in the radius of 10 m from source. The surface area, where the values obtained from the models differ above 25%, are in this case equal to 33%. For the other measurements in all other scenarios the 25% and less values difference was obtained in the surface area in range 15–33% of 25 m radius circle.

5 Conclusions and Future Work

The article presents the results of measurements and modelling of the performance of the jammers in the open area in LOS conditions. Based on the results of measurements it can be noticed that jammers behave differently for different frequency channels of WiFi band (Figs. 3, 4, 5, 6, 7 and 8). It occurs that shielding radius given by the manufacturer of the devices does not match the measurements. Effective range at −75 dBm is twice shorter. In all experiment scenarios the path loss models were calculated. Both models follow the log-distance rule, however log-normal model is better fitted at distances greater than 4 m and two-ray ground reflection one for smaller distances. Elaborated models allow for estimating the distribution of the signal power coming from jamming sources and the performance of the jammers' (Fig. 11). Results obtained from calculations fit quite well measurements. In the area of measurements (25 m radius circle) calculated values of signal power differs by maximum 25% in surface area range 85%–67% comparing to that obtained from Fluke XT analyser.

In the future work the authors plan to extend measurements for checking the jammers performance in the open area with working WiFi network containing a few (2–3) nodes. This will allow to estimate the band jamming profile in different channels and will help to locate and track the jamming source. This is very important issue for security in wireless networks.

References

1. Deng, S., Netravali, R., Sivaraman, A., Balakrishnan, H.: WIFI, LTE, or both?: Measuring multi-homed wireless internet performance. In: Proceedings of the 2014 Conference on Internet Measurement Conference, pp. 181–194. ACM (2014)
2. Gaj, P., Jasperneite, J., Felser, M.: Computer communication within industrial distributed environment-a survey. IEEE Trans. Ind. Inform. **9**, 182–189 (2013)
3. Aldlawie, A.H., QasMarrogy, G.A.: Performance evaluation of the effect of noise power jammer on the mobile bluetooth network. Int. J. Comput. Netw. Commun. (IJCNC) **6**(5), 167–174 (2014)
4. Punal, O., Pereira, C., Aguiar, A., Gross, J.: Experimental characterization and modeling of RF jamming attacks on VANETs. IEEE Trans. Veh. Technol. **64**(2), 524–540 (2015)
5. Pelechrinis, K., Koutsopoulos, I., Broustis, I., Krishnamurthy, S.: Jammer localization in wireless networks: an experimentation-driven approach. Comput. Commun. **86**, 75–85 (2016)
6. Cheng, T., Li, P., Zhu, S., Torrieri, D.: M-cluster and X-ray: two methods for multi-jammer localization in wireless sensor networks. Integr. Comput. Aided Eng. **21**(1), 19–34 (2014)
7. Aziz, F., Shamma, J., Stuber, G.L.: Jammer type estimation in LTE with a smart jammer repeated game. IEEE Trans. Veh. Technol. PP(99), 1 (2017)
8. Liu, G., Liu, J., Li, Y., Xiao, L., Tang, Y.: Jamming detection of smartphones for WiFi signals. In: IEEE 81st Vehicular Technology Conference (VTC Spring), pp. 1–3 (2015)
9. Mistry, H.P., Mistry, N.H.: RSSI based localization scheme in wireless sensor networks: a survey. In: Fifth International Conference on Advanced Computing and Communication Technologies, pp. 647–652, 21–22 February 2015
10. Kwiecień, A., Maćkowski, M., Kojder, M., Manczyk, M.: Reliability of bluetooth smart technology for indoor localization system. In: Gaj, P., Kwiecień, A., Stera, P. (eds.) CN 2015. CCIS, vol. 522, pp. 444–454. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19419-6_42
11. Czerwinski, D., Przylucki, S., Wojcicki, P., Sitkiewicz, J.: Path loss model for a wireless sensor network in different weather conditions. In: Gaj, P., Kwiecień, A., Sawicki, M. (eds.) CN 2017. CCIS, vol. 718, pp. 106–117. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59767-6_9
12. Ecer - Cell Mobile Phone Jammer Company, Cell Phone and Wifi Jammer [CRJ4000] Parameters, July 2016. http://www.ecer.com/products/cell_phone_and_wifi_jammer_crj4000-mpz234a62a-z1fcb0dd/showimage.html
13. CKJ-1502A12 Jammer, ChingKong Technology, July 2016. http://www.szckt.com/ckj-1502a12-p00079p1.html
14. Fluke Networks, Air Magnet Spectrum XT Analyser Datasheet. http://airmagnet.flukenetworks.com/assets/datasheets/AirMagnet_SpectrumXT_Datasheet.pdf. Accessed Dec 2017

15. Santos, P.M., Abrudan, T.E., Aguiar, A., Barros, J.: Impact of position errors on path loss model estimation for device-to-device channels. IEEE Trans. Wirel. Commun. **13**(5), 2353–2361 (2014)
16. Chen, J., Zhang, R., Song, L., Han, Z., Jiao, B.: Joint relay and jammer selection for secure two-way relay networks. IEEE Trans. Inf. Forensics Secur. **7**(1), 310–320 (2012)
17. Liu, Z., Liu, H., Xu, W., Chen, Y.: Error minimizing jammer localization through smart estimation of ambient noise. In: 2012 IEEE 9th International Conference on Mobile Adhoc and Sensor Systems (MASS), pp. 308–316 (2012)
18. Wei, X., Wang, T., Tang, C., Fan, J.: Collaborative mobile jammer tracking in Multi-Hop Wireless Network. Future Gener. Comput. Syst. **78**, 1027–1039 (2016)
19. Heath, R.W., Gonzalez-Prelcic, N., Rangan, S., Roh, W., Sayeed, A.M.: An overview of signal processing techniques for millimeter wave MIMO systems. IEEE J. Sel. Topics Sig. Process. **10**(3), 436–453 (2016)
20. Dhoutaut, D., Régis, A., Spies, F.: Impact of radio propagation models in vehicular ad hoc networks simulations. In: Proceedings of the 3rd International Workshop on Vehicular ad hoc Networks, pp. 40–49. ACM (2006)
21. Olasupo, T.O., Otero, C.E., Olasupo, K.O., Kostanic, I.: Empirical path loss models for wireless sensor network deployments in short and tall natural grass environments. IEEE Trans. Antennas Propag. **64**(9), 4012–4021 (2016)
22. Alsayyari, A., Kostanic, I., Otero, C.E.: An empirical path loss model for Wireless Sensor Network deployment in a concrete surface environment. In: 2015 IEEE 16th Annual Wireless and Microwave Technology Conference (WAMICON), Cocoa Beach, pp. 1–6 (2015)
23. Miranda, J., Abrishambaf, R., Gomes, T., Gonçalves, P., Cabral, J., Tavares, A., Monteiro, J.: Path loss exponent analysis in wireless sensor networks: experimental evaluation. In: 11th IEEE International Conference on in Industrial Informatics (INDIN), pp. 54–58 (2013)
24. Sommer, C., Eckhoff, D., German, R., Dressler, F.: A computationally inexpensive empirical model of IEEE 802.11 p radio shadowing in urban environments. In: Eighth International Conference on Wireless On-Demand Network Systems and Services (WONS), pp. 84–90. IEEE (2011)
25. Scott L.: Fast jammer detection and location using cell-phone crowd-sourcings. J 911: GPS World **21**(11) (2010)
26. Bayraktaroglu, E., King, C., Liu, X., Noubir, G., Rajaraman, R., Thapa, B.: Performance of IEEE 802.11 under jamming. Mob. Netw. Appl. **18**(5), 678–696 (2013)

LTE Load Balancing with Tx Power Adjustment and the Real Life Mobility Model

Konrad Połys(✉), Krzysztof Grochla, and Michał Gorawski

Institute of Theoretical and Applied Informatics, Polish Academy of Science,
Bałtycka 5, 44-100 Gliwice, Poland
{kpolys,kgrochla,mgorawski}@iitis.pl

Abstract. The load balancing in the LTE and LTE-A networks allows usage of the radio resources more effectively to improve client satisfaction. The paper provides analysis of the influence of complex user mobility patterns on the load balancing algorithm. The simulation models are used to evaluate the number of unsatisfied clients depending on the mobility model used. Two mobility models are used: RLMM and Random Waypoint. The Tx Power Adjustment method is evaluated, which is based on reconfiguration of a LTE eNodeB TX Power and was proposed by authors in their previous papers. Method proved to be effective and fairly easy in implementation, and the evaluation through series of simulation shown up to 50% rise in clients satisfaction with upholding the total network efficiency.

Keywords: LTE · LB · Load balancing · SON · HO · Handover
RLMM · Real Life Mobility Model

1 Introduction

With the rise of the LTE as a leading standard in nowadays cellular networks and the increasing activity in network traffic [1], there is a need to use the full spectrum available for an operator in each of the cells.

This leads to decreasing the cell size using multiple low power and low range base stations and in effect to increased intra-cell interferences. To deal with this problem the frequency allocation schemes are used (e.g. Fractional Frequency Reuse (FFR) and Soft Frequency Reuse (SFR) [2]).

In LTE, depending on the signal quality, different modulation and coding schemes (MCS) are used to define the amount of bits per radio resource blocs (RRB) which can be transmitted to and by a particular UE. At the same time provider has to assure, that the number of clients serviced with lower throughput is minimal with maximal amount of transferred data.

While moving around the network clients are usually serviced by several eNodeBs, if client is in the middle of a voice or data session, handover is realized

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 183–192, 2018.
https://doi.org/10.1007/978-3-319-92459-5_15

in which client is seamlessly patched from one base station to the other without loosing it's ongoing session. By default, client is connected to the base station with the highest value of Reference Signal Received Power (RSRP), however this approach can lead to overload of cells in more densely populated areas. The load balancing algorithms have to ensure the handover reliability with the optimization of the network resources usage.

There are several methods of dealing with load balancing in the LTE networks, usually they base on shifting the load between neighbouring eNodeBs. Sometimes moving the client from overloaded node to less occupied node with lower value of RSRP can improve the clients satisfaction. One method is to prefer handover to less populated nodes [3]. This relatively simple solution proven to be an effective method in reducing nodes overload. Several other solutions base on vertical handovers [4], hybrid decision approach for the association problem in heterogeneous networks [5], selection of the best radio access technology basing on game theory [6], dynamics of network selection in a heterogeneous wireless network [7], Cell Range Expansion approach which adds an offset to RSRP of pico cell [8].

The following paper focuses on a TX power adjustment method, presented in [9]. This solution regulates loads by changing the Tx power and adjusting the SFR parameters reducing the interferences [10].

The rest of the paper is organized as follows: second section is a short description of Tx power adjustment solution with pseudo code, Sect. 3 describes the Real Life Mobility Model used as a source of mobile nodes movement in simulation. Section 4 describes the simulation environment and the results of simulation with the results comparison form [9] and the RLMM model [11,12].

2 Tx Power Adjustment

The Tx power adjustment method presented in [9], bases on dynamic change of the base station transmission power in reaction to clients density with addition of the Soft Frequency Reuse technique.

The dynamic changes in Tx power lowers the Tx power of overloaded nodes on the outside of base station range and simultaneously increases the power in the inner area thus increasing UE's bandwith and sets clients that will loose the BS connection for handover. The BS selected to take over the "lost" clients raises it's Tx power to perform needed handovers.

Following the algorithm presented previously in [9]:

Variables description:

lLoadIn - the power of inner area for eNB with low load
lLoadOut - the power of outer area for eNB with low load
lLPc - point of change between inner and outer area for eNB with low load
hLoadIn - the power of inner area for eNB with high load
hLoadOut - the power of outer area for eNB with high load
hLPc - point of change between inner and outer area for eNB with high load
defIn - the power of inner area for eNB with default settings

defOut - the power of outer area for eNB with default settings
defPc - point of change between inner and outer area for eNB with default settings
eNB - single eNodeB
eNBs - list of all eNodeBs
crowdedeNBs - list of eNodeBs with high load
lleNBs - list of eNodeBs with low load
UEsTable - list of UEs connected to the current eNodeB
alternativeeNBsTable - list of eNodeBs which are not highly loaded and UEs could connect to them

```
setVariablesForPowers(lLoadIn, lLoadOut, lLPc,
hLoadIn, hLoadOut, hLPc, defIn, defOut, defPc)
setVariablesForLoad(overLoad, medLoad)
for each eNB in listOf(eNBs): eNB.getLoad()
    if eNB.getLoad() > overLoad: crowdedeNBs.put(eNB)
for each eNB in listOf(crowdedeNBs):
    UEsTable=eNB.getListOfConnectedUEs()
    for each UE in listOf(UEsTable):
        alternativeeNBsTable.put(UE.geteNBwith2ndBestReception)
    alternativeeNBsTable.removeeNBsWithLoadHigherThan(medLoad)
    lleNBs.put(alternativeeNBsTable.findMostPopular())
lleNBs.leaveOnlyUniqeeNBs()
for each eNB in listOf(lleNBs):
    eNB.setTxPowerInside(lLoadIn)
    eNB.setTxPowerOutside(lLoadOut)
    eNB.setPointOfChangeBetweenInsideOutside(lLPc)
for each eNB in listOf(crowdedeNBs):
    eNB.setTxPowerInside(hLoadIn)
    eNB.setTxPowerOutside(hLoadOut)
    eNB.setPointOfChangeBetweenInsideOutside(hLPc)
for each eNB in listOf(eNBS):
    if eNB is not in(crowdedeNBs and lleNBs):
        eNB.setTxPowerInside(defIn)
        eNB.setTxPowerOutside(defOut)
        eNB.setPointOfChangeBetweenInsideOutside(defPc)
```

Next, the rest of eNBs, which are not reconfigured in this round, have set their settings to default. In described scenario the following values were used: lLoadIn = 40 dBm, lLoadOut = 43 dBm, lLPc = 90%, hLoadIn = 38 dBm, hLoadOut = 39 dBm, hLPc = 80%, defIn = 37 dBm, defOut = 40 dBm, defPc = 60%, overload = 1.0, medLoad = 0.85. Values for high and low load were attained experimentally. We express the network load as a percentage of used radio resource blocks (RRB) among all RRBs in all cells in the network. Load equal 100% means that all RRBs were allocated.

The algorithm can be also described with following steps:

1. Get the load of all base stations and in case of overload (load above the given threshold) create the list of overcrowded eNodeBs - **overloadList**.
2. For each eNodeB (from **overloadList**) create a list of connected UEs - **connectList**.

3. Create a list of neighbouring eNodeBs (from **connectList**) where UEs will have the highest signal strength - **targetBSList**.
4. If the eNodeBs load is over the threshold, remove it from the **targetBSList** list (creating the list of base station's neighbours with free resources - **freeList**).
5. From the **freeList** select eNodeB that can service excess clients from overcrowded node.
6. Reconfigure the transmission parameters of overloaded base stations and their neighbours. Lower the transmit power for overloaded eNodeBs and increase the transmit power for chosen neighbours. Changes are made with consideration of SFR.

Simulation consists of 19 LTE base stations (eNodeB) and number of UEs. We use the honeycomb topology, with area of simulation scenario 14.9×13.3 km. Simulation is implemented in OMNeT++ with INET framework. In [9] authors tested the Random Waypoint mobility model [13] as a source of movement. In the following paper the RLMM mobility model [11,12] is added and compared with the previous results

3 Mobility Modeling - The Real Life Mobility Model

The LTE network, just like any cellular network deals mainly with two kinds of mobility source. The moving devices are mostly mobile devices carried by people travelling by foot or using some kind of vehicle (public transport, car, bicycle etc.). Vehicles move using public roads with set of rules governing the traffic, while people movement is much less restricted. However what most working people have in common is the cyclic character of their actions during the day, and the need for sleep during the night. The standard mobility models (e.g. random waypoint) used in simulations of mobility in cellular networks does not include the cyclic behaviour of cellphone users, thus does not present actual changes in the network loads.

The RLMM mobility model proposed in [11,12] simulates movement of a working person, basing on day/night and weekday/weekend cycles.

The movement between points of interests (home, work, hotspot etc.) is realised in a semi random manner to assure some deviations from a straight point-to-point travel (Fig. 1).

The experiments with RLMM [11,12] shows it's characteristic differences from usually used mobility models and creates the clear division on cycles in UEs activity. The night/day cycles show the difference in loads of eNodesB placed in the housing areas and work areas. Model also captures the fact that according to [14] people in developed country travel only up to 2 h a day, and spend most of they working day around 2 places (eNodesB) - work and home locations (there are exceptions of course, the distances covered by analyzed traces follows fat-tailed distribution so while most people travel around 1–10 km per day certain individuals have jobs that require much longer travels).

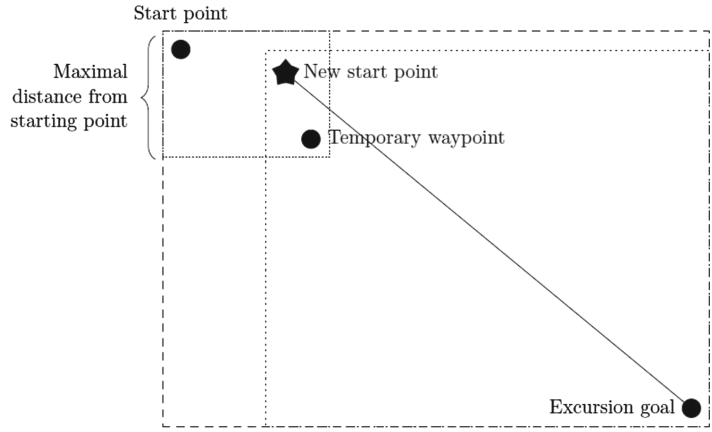


Fig. 1. RLMM choosing semi-random travel path [11]

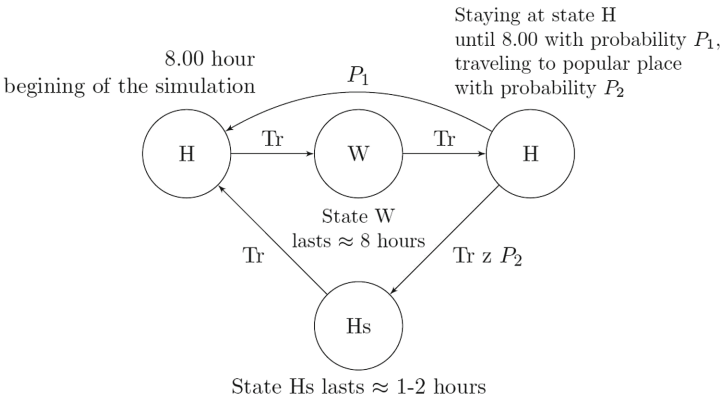


Fig. 2. RLMM model weekday activity [12]

4 Simulation Model Results

For the simulation purposes we used the simulation already prepared in [9] and run it with the movement traces generated from RLMM model.

The RLMM model was run for 100, 200 and 300 UEs for period of 21 days. Model was set on standard 7 days week with 5 workdays and 2 days weekend.

Weekdays set the UEs to start day at their defined home place, wake up, travel to work, and after work return home. After work the UE stays home until next iteration with probability P1 or it will travel to a popular place (hotspot,

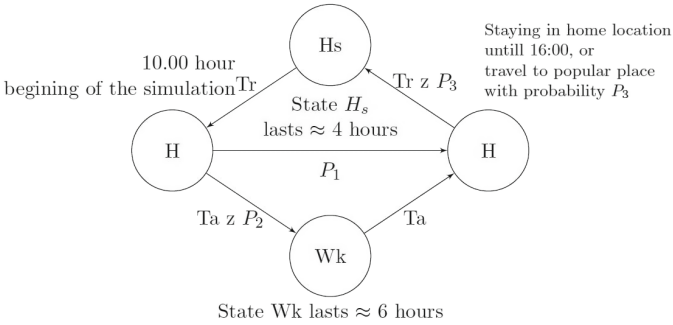
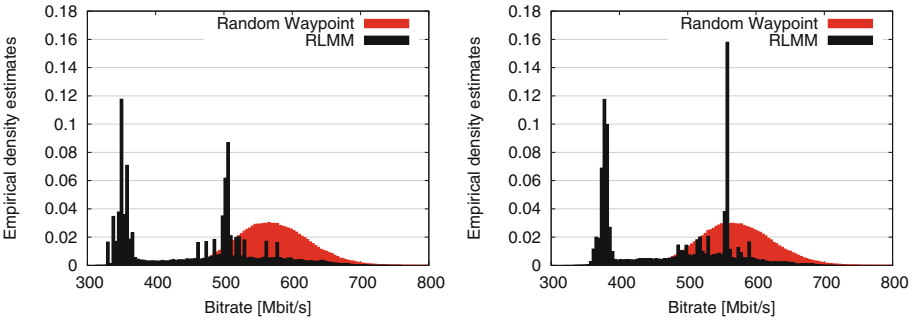


Fig. 3. RLMM model weekend activity [12]

e.g. bar, shopping mall, cinema etc.) for some time before sleep with probability P2. As default P2 is set to 100% (Fig. 2).

During weekends wake up time is set to around 2 h after value set during weekdays, the simulated UE stays home with probability P1 or travels to weekend activity site with probability P2. After returning from weekend activity UE can travel to evening activity hotspot with probability P3 (longer time than during weekdays). As default P2 and P3 are set to 100% (Fig. 3)



(a) without algorithm (b) with algorithm

Fig. 4. Network bitrate comparison between Random Waypoint and RLMM (200 UEs)

The difference in mobility models characteristics can be observed on Fig. 4, where the example of bitrate probability is presented for the network with both mobility models implemented for 200 UEs. The red graph presents the random waypoint model, for which the implementation of balancing algorithm did not change much concerning total throughput. For the RLMM model (black) we can observe more visible changes in offered throughput. The Fig. 5 shows the bitrate change in time during the course of one week using the RLMM model. As

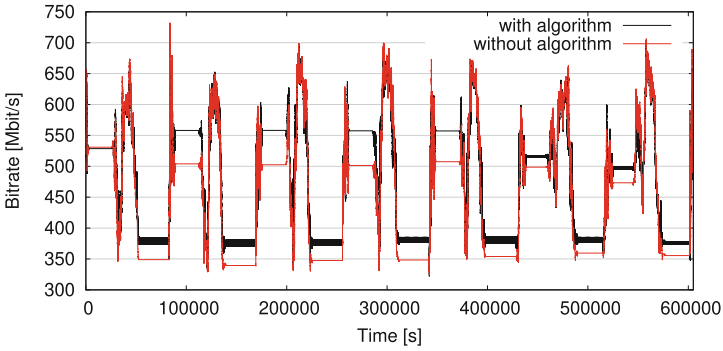


Fig. 5. Network bitrate during 1 week (200 UEs)

expected the model characteristics can be observed through available throughput with the phases of UEs moving and stationary.

The analysis done in [9] proven that the balancing algorithm is very effective while the network is not overloaded or when only parts of the network are overcrowded.

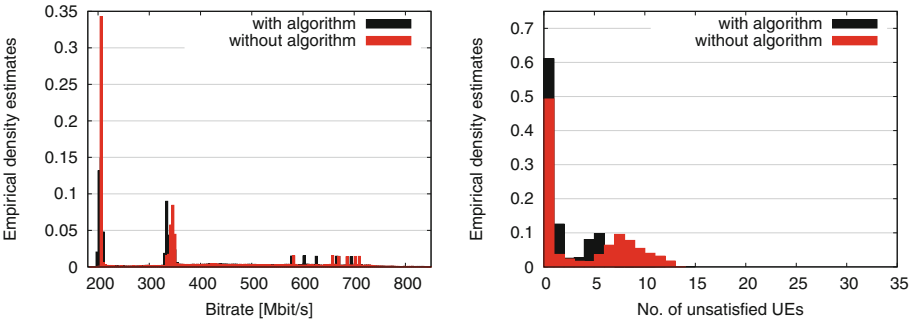
On the below figures we can observe that the course is highly bonded with the mobility model that is used. Sum of throughput and the count of unsatisfied clients (situation when at least 512 Kbit/s throughput is not met) changes during mobility. With the 100 devices simulated, in simulation without any algorithms improving received signal or throughput, average throughput is 379.92 Mbit/s (Fig. 6a red) and the average of unsatisfied clients is 3.49 (Fig. 6b red).

For the results with the Tx Power solution applied average throughput is 376.61 Mbit/s (Fig. 6a black) and average unsatisfied level is only 1.35 (Fig. 6a black). The average throughput is lower because of more fair resources allocation and the count of simulated devices is relatively small.

When the count of simulated devices is increased to 200 the increase of satisfied clients is much more visible. Much less clients are classified as unsatisfied, and higher average throughput is obtained. Without any improving algorithms the simulation with 200 devices showed average thorughput equal to 453.20 Mbit/s (Fig. 7a red) and 14.75 unsatisfied clients (Fig. 7a red). The same simulation with presented algorithm gives 475.14 Mbit/s average throughput (Fig. 7a black) and 0.73 unsatisfied clients (Fig. 7a black). Third simulated case with same simulation area but with 300 devices provides next results which verifies presented method. Third case without improving algorithms results in 596.13 Mbit/s of average throughput (Fig. 8a red) and 14.89 (Fig. 8a red) of average unsatisfied clients. When described algorithm is applied then average throughput is equal 601.59 Mbit/s (Fig. 8a black) and 5.70 (Fig. 8a black) as average unsatisfied clients count.

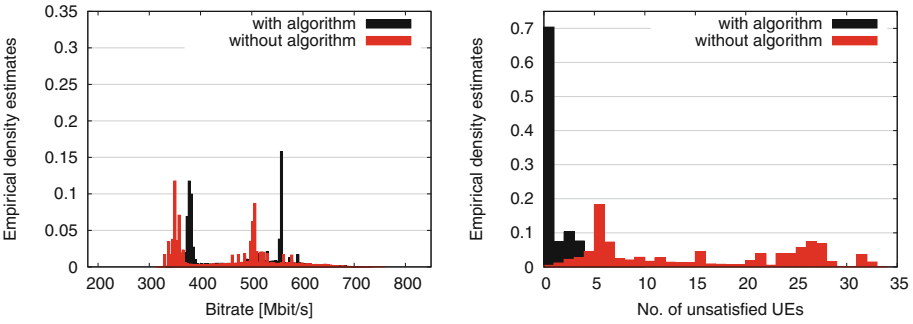
Table 1. Summarized results

Number of devices	Throughput w/o improving algorithms	Throughput with optimization	Unsatisfied clients w/o improving algorithms	Unsatisfied clients with optimization
100	379.92 Mbit/s	376.61 Mbit/s	3.49	1.35
200	453.20 Mbit/s	475.14 Mbit/s	14.75	0.73
300	596.13 Mbit/s	601.59 Mbit/s	14.89	5.70



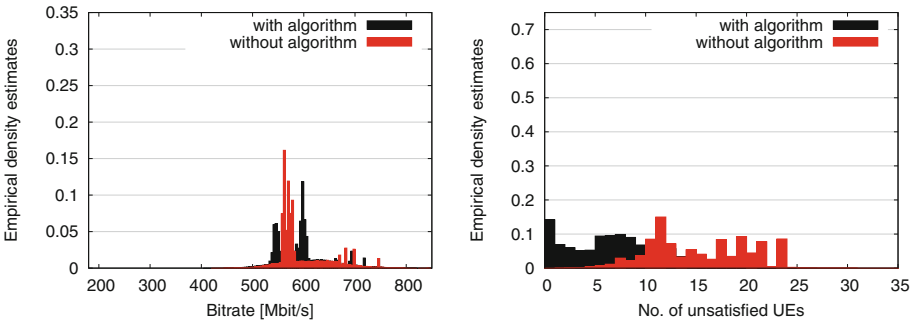
(a) network bitrate (b) unsatisfied clients

Fig. 6. Histograms of network with 100 UEs (Color figure online)



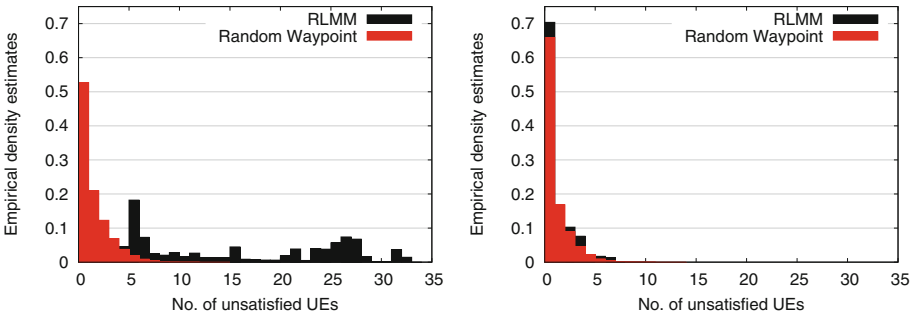
(a) network bitrate (b) unsatisfied clients

Fig. 7. Histograms of network with 200 UEs (Color figure online)



(a) network bitrate (b) unsatisfied clients

Fig. 8. Histograms of network with 300 UEs (Color figure online)



(a) without algorithm (b) with algorithm

Fig. 9. No. of unsatisfied UEs comparision between Random Waypoint and RLMM (200 UEs)

5 Conclusion

The paper continues the research on a balancing method presented in [9]. The new experiments presented in the paper shows the behaviour of a load balancer while dealing with more realistic mobility than in previous tests. The characteristics generated using the RLMM mobility model widely differ from the ones presented in [9] (Fig. 9). The summary Table 1 presents the results of experiments for 100, 200 and 300 UEs. The results clearly shows that the radical change of the mobility characteristics did not change the fact, that the balancing algorithm manages still to improve the clients satisfaction by a large margin.

References

1. Cisco: Cisco visual networking index: global mobile data traffic forecast update, 2013–2018, Cisco Public Information (2014)
2. Boudreau, G., Panicker, J., Guo, N., Chang, R., Wang, N., Vrzic, S.: Interference coordination and cancellation for 4G networks. IEEE Commun. Mag. **47**(4), 74–81 (2009)
3. Lobinger, A., Stefanski, S., Jansen, T., Balan, I.: Coordinating handover parameter optimization and load balancing in LTE self-optimizing networks. In: 2011 IEEE 73rd Vehicular Technology Conference (VTC Spring), pp. 1–5. IEEE (2011)
4. Stevens-Navarro, E., Lin, Y., Wong, V.W.: An MDP-based vertical handoff decision algorithm for heterogeneous wireless networks. IEEE Trans. Veh. Technol. **57**(2), 1243–1254 (2008)
5. Elayoubi, S.E., Altman, E., Haddad, M., Altman, Z.: A hybrid decision approach for the association problem in heterogeneous networks. In: INFOCOM, 2010 Proceedings IEEE, pp. 1–5. IEEE (2010)
6. Aryafar, E., Keshavarz-Haddad, A., Wang, M., Chiang, M.: RAT selection games in HetNets. In: INFOCOM, 2013 Proceedings IEEE, pp. 998–1006. IEEE (2013)
7. Niyato, D., Hossain, E.: Dynamics of network selection in heterogeneous wireless networks: an evolutionary game approach. IEEE Trans. Veh. Technol. **58**(4), 2008–2017 (2009)
8. Damnjanovic, A., Montojo, J., Wei, Y., Ji, T., Luo, T., Vajapeyam, M., Yoo, T., Song, O., Malladi, D.: A survey on 3GPP heterogeneous networks. IEEE Wirel. Commun. **18**(3), 10–21 (2011)
9. Grochla, K., Połys, K.: Load balancing in LTE by Tx power adjustment. In: Gaj, P., Kwiecień, A., Stera, P. (eds.) CN 2016. CCIS, vol. 608, pp. 216–225. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39207-3_19
10. Grochla, K., Połys, K.: Dobór parametrów mechanizmu zwielokrotnienia wykorzystania częstotliwości sfr w sieciach lte. In: Materiały konferencyjne Krajowego Sympozjum Telekomunikacji i Teleinformatyki. KSTiT (2015)
11. Gorawski, M., Grochla, K.: The real-life mobility model: RLMM (2013)
12. Gorawski, M.: Load balancing and processing data from Internet of Things smart devices. Studia Informatica **38**(4) 2017
13. Gorawski, M., Grochla, K.: Review of mobility models for performance evaluation of wireless networks. In: Gruca, A., Czachórski, T., Kozielski, S. (eds.) Man-Machine Interactions 3, pp. 567–577. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-02309-0
14. Song, C., Qu, Z., Blumm, N., Barabási, A.-L.: Limits of predictability in human mobility. Science **327**(5968), 1018–1021 (2010). http://science.sciencemag.org/content/327/5968/1018

Determining the Usability of Embedded Devices Based on Raspberry Pi and Programmed with CODESYS as Nodes in Networked Control Systems

Jacek Stój(✉), Ireneusz Smołka, and Michał Maćkowski

Institute of Informatics, Silesian University of Technology,
Akademicka 16, 44-100 Gliwice, Poland
{jacek.stoj,ireneusz.smolka,michal.mackowski}@polsl.pl

Abstract. Since introduction of the Raspberry Pi embedded computers to the market, its popularity is constantly growing. More and more both hardware and software modules are available for that device. One of the later ones is CODESYS development environment, which make it possible to program Raspberry Pi like a regular Programmable Logic Controllers PLC. Using its network interfaces and industrial protocols like Ethercat or Profinet available in CODESYS, too, one could consider using Raspberry as a node in Networked Control System. Posts and opinions shared on Internet forums proofs it. However, one thing should not be ignored – Raspberry has been never intended to be implemented in industrial environment and was not designed as a real-time device. Therefore before practical application of Raspberry Pi, its temporal characteristics should be analyzed. It is the main concern in the following chapter.

Keywords: Raspberry PI · Real-time · Industrial system
Networked control system · CoDeSys · Arduino · LabView
Embedded devices

1 Introduction

In recent years it could be observed that more and more various embedded devices are available on the market. They are supplied by many vendors in numerous configurations referring to their hardware components – integrated or available as options. As a result embedded devices are implemented in many areas in great range of different applications like building automation [1,2], data processing [3], image acquisition processing [4], distributed computer systems [5] or Internet Of Things [6] to name only few. Generally speaking, embedded devices are most often used in Cyber-Physical Systems.

The above examples refer to one of the most recognizable embedded devices – Raspberry Pi. That specific device is unique and worth mentioning also because

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 193–205, 2018.
https://doi.org/10.1007/978-3-319-92459-5_16

of the possible ways of programming it. The Raspberry Pi programmer can use basic programming languages like Python or C. Independent companies (e.g. Microsoft, Google, Ubuntu) offer their own operating systems and let create programs in languages like Java or C#. Another option of Raspberry Pi programming is CODESYS - Controller Development System.

CODESYS development environment is dedicated to automation applications for industrial controllers. It is the base of many other development environments, which are used for programming Programmable Logic Controllers, e.g. TwinCAT by Beckhoff, Easy Soft by Eaton (formerly Moeller). Using CODESYS for programming Raspberry Pi makes it operates like regular PLC. The program is being realized in a never ending loop. Each program realization is called cycle. It is preceded by inputs acquisition and followed by outputs update. Implementation of the user program is possible according to international standard IEC 61131 which, among other things, defines programming languages dedicated to PLCs (LD, FBC, ST, IL, SFC [7,8]). Finally, Raspberry Pi together with CODESYS is not only about programming and operation. Many industrial communication protocols are available: EtherCAT, Profinet, Modbus TCP, OPC UA for creation of Networked Control Systems NCS [9]. It makes Raspberry Pi worth considering for real applications.

However, the question arises - if Raspberry Pi may operate like a PLC, couldn't it be used like one? Can we implement Raspberry in Networked Control Systems? At this point another feature of NCS node should be pointed out, which is temporal determinism [10,11]. Most of the NCS nodes operate with real-time requirements. Their proper operation depends not only on right calculation results, but also on the time when the results are obtained. Can Raspberry Pi keep this requirement? In the paper authors will try to give an answer to that question.

This paper is structured as follows: the next section presents the related work. Section 3 includes other Raspberry Pi applicability considerations not associated in its operation in the time domain. In Sect. 4 the testbed is described, followed by experimental research results in Sect. 5. The last section contains conclusions and final remarks.

2 Related Work

As mentioned in the previous section, there are many application examples of Raspberry in computer systems, and of great variety, too. However, the Raspberry usually gives some non-critical functionality like in GPS module coupled with a Raspberry PI as data logger in [12], detecting objects in surveillance area [13] or assessing fishing vessel stability [14].

Some of them are also associated with real-time systems. In [15] a system is described in which Raspberry Pi equipped with camera was used for obstacle detection. The average image processing performed by Raspberry was measured. However, the processing was not considered from the point of view of satisfying real-time system constraints. Another example is the development of CAN bus

communication interface for Raspberry in order to monitor vehicle parameters [16]. The project was finished successfully, but the conclusion was quite general and said the CAN interface "works". There were no real-time considerations, either. It is similar in other works that were available to authors like [17–19].

There are pieces of information in the Internet about some measurements performed on Raspberry Pi that were supposed to show how fast Raspberry can react to requests, e.g. signals of what frequency could be generated on its output. The issue of real-time capability of Raspberry is also raised on Internet forums where its applicability in industry is considered. However, this paper, to the best of our knowledge, is the first that describes the experimental results of the response time of Raspberry Pi to incoming requests.

3 Other Applicability Considerations

Before using Raspberry Pi boards and related based hardware for the industrial applications, it is necessary to consider the issue of its environmental characteristics. One thing is the heat generated by the device during operation. Under load the Raspberry PI may get hot and it should be considered whether it is acceptable in given application, especially in industrial environments.

Another thing is the EMC issues. On one side, the system has to be resistant to the electromagnetic disturbances occurring in the industrial environments, and on the other it should not be the source of high level electromagnetic emission. According to the Raspberry Pi manufacturer, all models are tested for CE compliance in a "typical configuration" and conform to various standards harmonized with EU Directives such as: Radio Equipment 2014/53/EU, Electromagnetic Compatibility (EMC) 2014/30/EU. In cases when the apparatus can take different configurations, the EMC assessment confirms that the apparatus meets the protection requirements in all possible configurations identified by the manufacturer as representative of its intended use. In practice, in such universal product like Raspberry Pi, it is almost impossible to find the worst case of EMC scenario. According to the EMC Directive [20] and Guide for the EMC Directive [21], when a manufacturer assembles a final apparatus using components from other company, the manufacturer must retain overall control.

Another issue is the impact of the application executed by the embedded system to the level of conducted and radiated emissions. According to the research results presented in papers [22–24] it can be notice that different program realizing the same functionality, clock sources, and programmable PLL (Phase Locked Loop) settings can be the source of different EMI (Electromagnetic Interference) level. The software can be also used to control the EMI level of the entire system as in papers [25–27], where the authors focus on software-level technique EMI optimization.

To summarize, each manufacturer which uses the Raspberry Pi as a part of the system is responsible for the compliance of the product with EU Directives. This is especially important when the board is used in industrial real time application, and which the safety and correct operation of other devices may depend on.

4 Testbed

The goal of the experimental research was to determine how much one could rely on the Raspberry Pi operation considered from the point of view of its temporal characteristics. We used Raspberry Pi version 3 programmed with CodeSys development environment so the user program implemented in the device was being executed like in a typical Programmable Logic Controller PLC. The point of interest was the time needed for the program execution.

The research was performed with original Raspberry Pi Foundation OS – Raspbian. The OS was in the "lite" version, without classical desktop and applications which are by default installed in regular version. The CodeSys program is implemented on a PC and then sent to Raspberry Pi to be executed in CodeSys runtime environment.

The user program may be configured to be executed in a couple of operation modes – cyclic, freewheeling, event triggered, status triggered. In this paper we focus on first two, as they are also typical to PLCs. In the cyclic execution mode the time between every execution of the user program is constant and may be defined by the user. In freewheeling mode the program execution starts one after another, i.e every execution starts after the previous execution is finished.

The research was performed in four different hardware configurations as described in the following subsections. In some cases we were also changing the length of the user program. Each time we measured the delay from one execution of the user program to another. In industrial environments it is associated with the response time of the device, which is crucial from the point of view of its real-time characteristics.

4.1 Generator and Oscilloscope

The first method which we proposed was based on a generator and oscilloscope – G&O method (see: Fig. 1). The generator was configured to generate a 1 kHz square wave. This signal was connected to the input of Raspberry Pi. Program in Raspberry Pi reacted to the change of the state of the input by toggling one of its outputs. As a result, from the Raspberry Pi we got a signal like the generated one, but with some delay. The length of the delay depended on the duration of the Raspberry Pi user program execution, which was configured to cyclic with 100 µs cycle time. The same experiment for the cycle time of the Raspberry Pi equal 1 ms was performed, too. The results were analogous. The measured delay was between 1 ms and 2 ms.

The consecutive durations of the time shifts between the generated square wave and the signal received from the Raspberry was measured by an oscilloscope and the results were stored in a CSV file. All of the measured values were in the range from 100 µs to 200 µs, which corresponds to our expectations – the minimum response time of devices operating in cyclic mode is equal one cycle time, while the maximum response time is not more than two cycles duration [28].

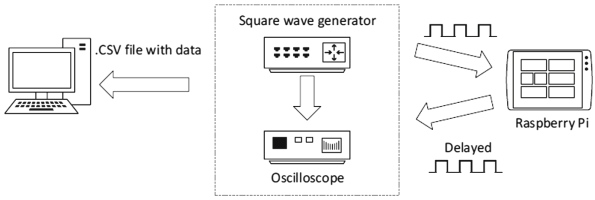


Fig. 1. Testbed based on a square wave generator and oscilloscope.

The results are shown in Fig. 2. On the x-axis the measured time is given. The y-axis shows the number of measurements (samples) with the given time. For example, there were about 500 samples with the measured time in the range 180 to 200 μs, over 600 samples (the highest bar on the graph) with the measured time in the range 160–180 μs, etc. There were also some samples with the time slightly below 100 μs, but we considered them as a measurement error.

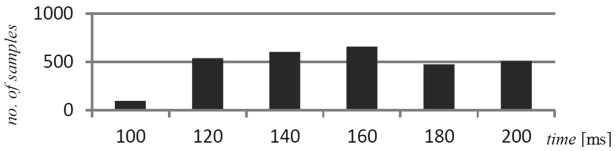


Fig. 2. Experiment results queried with generator and oscilloscope

In this experiment we got results corresponding to ours expectations. However, this method was used only as a way of validation of the following experiments. It gave reliable results, but in small quantity. To get the above results, the experiment had to be repeated many times. We performed 18 experiments with together about 3000 measurements (samples). To increase the number of measurements we prepared second method based on Data Acquisition Module DAQ by National Instruments.

4.2 Data Acquisition Module

In the second method, National Instruments Data Acquisition Module DAQ was used. The DAQ module connected by USB interface with PC computer, gave us the possibility to generate a square wave of configured frequency straight from PC, while performing the experiment just like the one described above. In this case however, the results could be stored on the PC without actual data size restrictions. We hoped that it could give us the possibility to perform long lasting experiments during which we could spot all deviations (if any were present) in the Raspberry Pi operation.

Unfortunately, using this method we got accuracy of 30 ms in comparison to G&O method (see: Fig. 4) because we didn't use LabView Real Time add-on

which we had no access to. In our research we wanted to have the accuracy of 1 ms so we decided to change measurement method again.

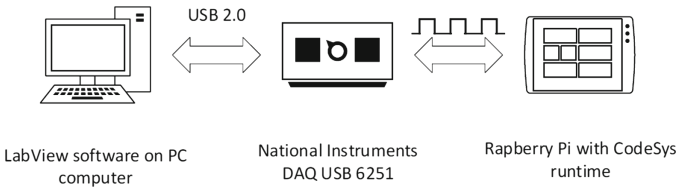


Fig. 3. Testbed based on a national instruments acquisition module

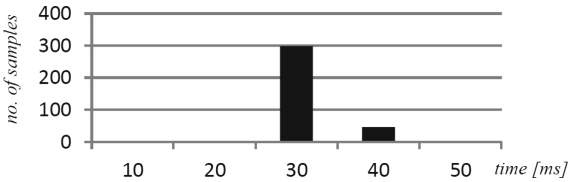


Fig. 4. Experiment results queried with NI acquisition module

4.3 ATmega Based Experiments – Request-Response Principle

Third method is similar to the G&O method, but instead of a generator and oscilloscope an ATmega microcontroller was used. It generated high level state signal on its output connected to the Raspberry Pi input. The Raspberry user program copied the state of that input onto its output, which was connected back to the ATmega input. This way the ATmega could determine the time needed for data processing by the Raspberry. After recording this measurement ATmega again changed the state of its output to low level starting another measurement.

The acquired results were not as expected. It was caused by "synchronization" of the ATmega and Raspberry Pi devices while performing the request-response transaction. It caused the response time to be irregular in the acquired time span. In Fig. 6 an example of those irregularities is shown. It was similar to the observations described in [29] and a result of convolution of the operation cycles of ATmega and Raspberry Pi devices. In the end, we modified the testbed once again as described in the following subsection.

4.4 ATmega Based Experiments – Square Wave Generator

In the last method described here, the Raspberry was changing the logical state of one of its outputs. The duration of low or high level of the output signal

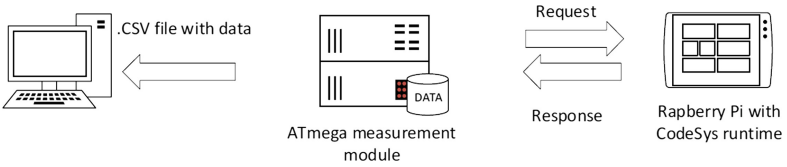


Fig. 5. ATmega based experiment based on the request-response principle

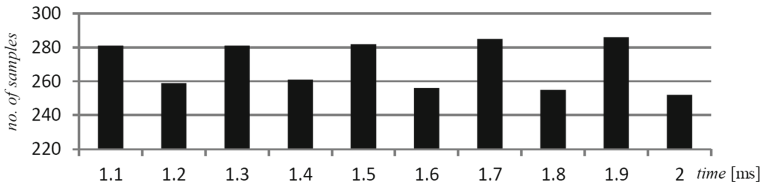


Fig. 6. Results acquired with the request-response method

showed the duration of the user program. The output was connected to an input of the ATmega which measured the time of changing of the logical signal level (Fig. 7).

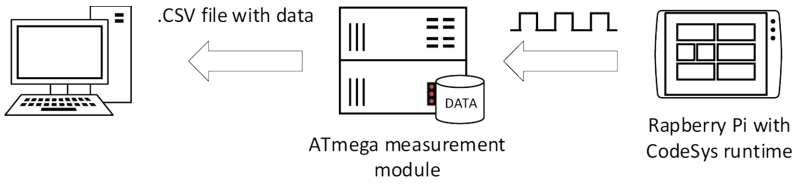


Fig. 7. ATmega and square wave generated by Raspberry Pi

The results were recorded in the ATmega internal memory. Comparison with the G&O results showed that they are reliable. That method was used for final experimental research described in the following section.

5 Results

In our research we tested how Raspberry Pi operates in cyclic mode and free-wheelin mode. We compared the temporal characteristics of updated of the Raspberry outputs which showed us how the user program was being executed. Additionally, we built a networked system based on the Modbus TCP protocol. The other network devices were five Arduino development boards with Ethernet shields. The Raspberry Pi was exchanging packets of 125 bytes data with the Arduinos every 100 ms.

5.1 Single Node System

First of all, we considered operation of Raspberry PI as a single node system. The user program implemented in Raspberry PI needed about 0.3 ms for single execution. We measured the Raspberry Pi cycle time RCP, i.e. the time from one user program execution to another. Two operation modes were checked – cyclic (with cycle time set to 0.5 ms) and freewheeling. The results are shown in Fig. 8.

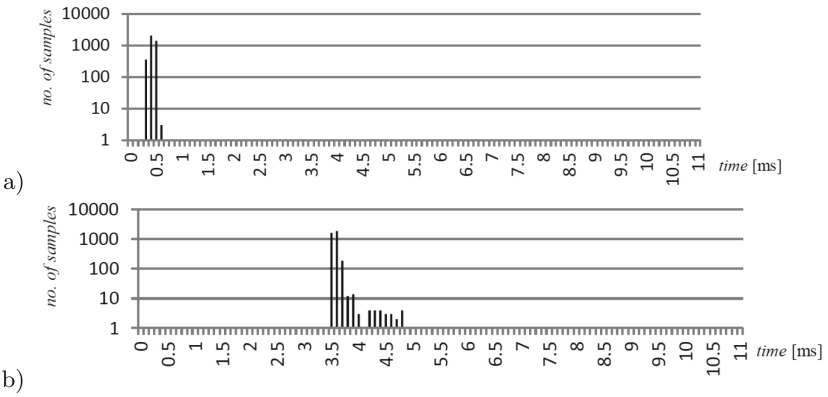


Fig. 8. Raspberry Pi cycle in single node system: (a) cyclic operation, (b) freewheeling operation

The above additional 3 ms in freewheeling operation mode was quite surprising. That operation mode is expected to be more efficient than cyclic mode, as in cyclic mode after every user program execution, the operating system waits for the cycle time to elapse. Whereas in freewheeling mode, the consecutive executions of the user program are to be realized on after another. Therefore, the RCP with different user program lengths was checked, too (see: Fig. 9).

In the performed measurements we confirmed that in freewheeling mode execution program is longer than the same program in cyclic mode. Additional time in freewheeling mode is about 3 ms and does not depend on the duration of the user program execution.

5.2 Networked System

To test the Raspberry Pi operation with active communication interface, Ethernet interface was used with Modbus TCP protocol. The network included five Arduino modules. With every Arduino one data exchange was being performed every 100 ms for reading 125 bytes of data. In this configuration we took 10,000 measurements. The results are show in Figs. 10, 11 and 12 (the y axis is scaled logarithmically).

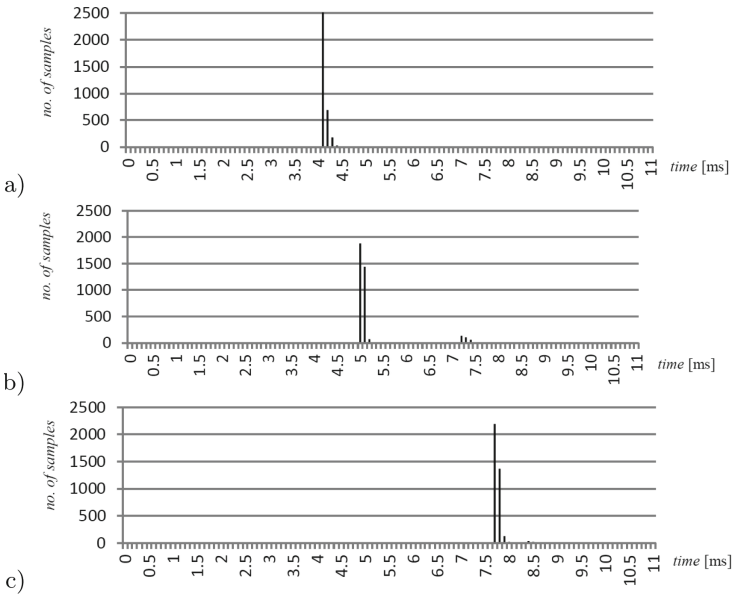


Fig. 9. Raspberry PI cycle times in freewheeling operation mode with different user program length: (a) 1 ms, (b) 2 ms, (c) 5 ms

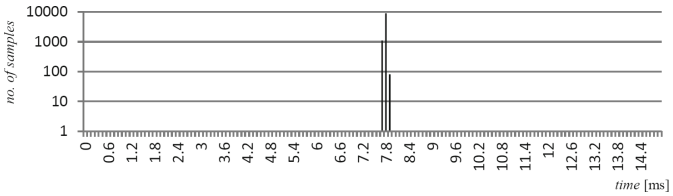


Fig. 10. The Raspberry PI cycle times in freewheeling operation mode without using the communication interface

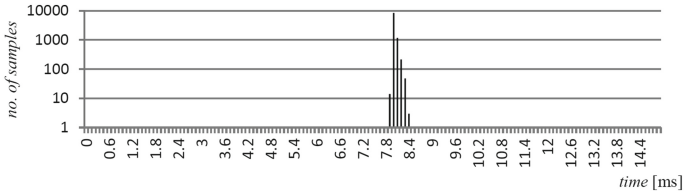


Fig. 11. The Raspberry PI cycle times in freewheeling operation mode with active Modbus TCP data exchange

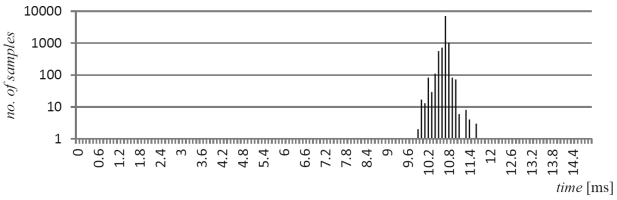


Fig. 12. The Raspberry Pi cycle times with active Modbus TCP data exchange

In case when Raspberry Pi run in freewheeling mode and is realizing Modbus TCP communication, the cycle time increases by about 1 ms. Interestingly, execution of the same user program in cyclic mode increases the cycle time to value greater than 10 ms - by more than 2 ms comparing to freewheeling mode.

6 Conclusions and Future Work

The paper presents some analysis and experimental results on the temporal aspect of Raspberry Pi operation programmed with CODESYS development environment. Two operation modes were examined – cyclic and freewheeling from the point of view of the user program execution time.

Working in freewheeling operating mode gives longer response times than in the cyclic operation mode for a not networked device. It had greater jitter, too. It is the behavior which is not intuitive. Skimming through Internet discussion forums suggests, that CODESYS programmer use freewheeling operation mode when want to achieve high responsiveness of the devices they program. The experiments show that it may have the opposite result.

The above remark seems not to apply to the operation of networked devices. When using communication interface (like the tested Modbus TCP), the Raspberry Pi had shorter response times in freewheeling mode than in cyclic mode. However, the difference in the response times is less significant.

The above considerations refer to the usage of Raspberry Pi programmed in CODESYS and therefore operating like regular PLC devices. However, its possible actual usage depends not only on its operation and temporal characteristics, but also on other issues like requirements for compliance testing as mentioned in Sect. 3. Nevertheless, we claim that the Raspberry Pi could be used in industrial environments for realization only non-critical task like system monitoring use for example for detection of improper operation. The programmer should also test every created configuration. Even little changes may have severe influence on the device operation.

Programming Raspberry Pi with CODESYS does not make a real-time device out of it. It is caused basically by the fact, that its operating systems is not of the real-time type. It is worth noticing however, that on the market there are embedded devices delivered by well-known vendors and used like regular PLC's which are also programmed with CODESYS and has non real-time operating

systems, either. Still, according to authors experience, they are used in industry for realization of networked control systems.

The research on the operation of devices programmed with CODESYS is planned to be continued. In future works the usage of another communication protocols like EtherCAT, Profinet will be analyzed and tested from the point of view of the influence on the device response time.

References

1. Jain, S., Vaibhav, A., Goyal, L.: Raspberry Pi based interactive home automation system through E-mail. In: 2014 International Conference on Reliability Optimization and Information Technology (ICROIT), pp. 277–280, February 2014
2. Vaidya, B., Patel, A., Panchal, A., Mehta, R., Mehta, K., Vaghasiya, P.: Smart home automation with a unique door monitoring system for old age people using python, opencv, android and raspberry Pi. In: 2017 International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 82–86, June 2017
3. Frieslaar, I., Irwin, B.: Investigating the electromagnetic side channel leakage from a raspberry Pi. In: 2017 Information Security for South Africa (ISSA), pp. 24–31, August 2017
4. Sharma, J., Anbarasu, M., Chakraborty, C., Shanmugasundaram, M.: Iris movement based wheel chair control using raspberry Pi - a state of art. In: 2017 Innovations in Power and Advanced Computing Technologies (i-PACT), pp. 1–5, April 2017
5. Wardi, Achmad, A., Hasanuddin, Z.B., Asrun, D., Lutfi, M.S.: Portable IP-based communication system using raspberry Pi as exchange. In: 2017 International Seminar on Application for Technology of Information and Communication (iSemantic), pp. 198–204, October 2017
6. Kadiyala, E., Meda, S., Basani, R., Muthulakshmi, S.: Global industrial process monitoring through IoT using raspberry Pi. In: 2017 International Conference on Nextgen Electronic Technologies: Silicon to Software (ICNETS2), pp. 260–262, March 2017
7. IEC 61131–3 Programmable controllers - Part 3: Programming languages (2003)
8. Rzońca, D., Sadolewski, J., Stec, A., Świder, Z., Trybus, B., Trybus, L.: CPDev engineering environment for control programming. In: Mitkowski, W., Kacprzyk, J., Oprzędkiewicz, K., Skruch, P. (eds.) Trends in Advanced Intelligent Control, Optimization and Automation. KKA 2017. Advances in Intelligent Systems and Computing, vol. 577, pp. 303–314. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-60699-6_29
9. Rząsa, W., Rzonca, D.: Event-driven approach to modeling and performance estimation of a distributed control system. In: Gaj, P., Kwiecień, A., Stera, P. (eds.) CN 2016. CCIS, vol. 608, pp. 168–179. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39207-3_15
10. Basile, F., Chiacchio, P., Gerbasio, D.: On the implementation of industrial automation systems based on PLC. IEEE Trans. Autom. Sci. Eng. **10**(4), 990–1003 (2013)
11. Jamro, M., Rzonca, D.: Impact of communication timeouts on meeting functional requirements for IEC 61131-3 distributed control systems. Automatika **56**(4), 499–507 (2015)

12. Masino, J., Frey, M., Gauterin, F., Sharma, R.: Development of a highly accurate and low cost measurement device for field operational tests. In: 2016 IEEE International Symposium on Inertial Sensors and Systems, pp. 74–77, February 2016
13. Menezes, V., Patchava, V., Gupta, M.S.D.: Surveillance and monitoring system using raspberry Pi and simplecv. In: 2015 International Conference on Green Computing and Internet of Things (ICGCIoT), pp. 1276–1278, October 2015
14. Abankwa, N., Squicciarini, G., Johnston, S., Scott, M., Cox, S.J.: An evaluation of the use of low-cost accelerometers in assessing fishing vessel stability through period of heave motion, October 2016
15. Mane, S.B., Vhanale, S.: Real time obstacle detection for mobile robot navigation using stereo vision. In: 2016 International Conference on Computing, Analytics and Security Trends (CAST), pp. 637–642, December 2016
16. Salunkhe, A.A., Kamble, P.P., Jadhav, R.: Design and implementation of can bus protocol for monitoring vehicle parameters. In: 2016 IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT), pp. 301–304, May 2016
17. Sahitya, S., Lokesha, H., Sudha, L.K.: Real time application of raspberry Pi in compression of images. In: 2016 IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT), pp. 1047–1050, May 2016
18. Joardar, S., Chatterjee, A., Rakshit, A.: A real-time palm dorsa subcutaneous vein pattern recognition system using collaborative representation-based classification. IEEE Trans. Instrum. Measur. **64**(4), 959–966 (2015)
19. Patruno, C., Marani, R., Nitti, M., D'Orazio, T., Stella, E.: An embedded vision system for real-time autonomous localization using laser profilometry. IEEE Trans. Intell. Transp. Syst. **16**(6), 3482–3495 (2015)
20. Guide for the EMC directive 2004/108/EC - european commission (2004)
21. Electromagnetic compatibiliy directive 2014/30/EU (2014)
22. Yuan, S.Y., Chung, W.Y., Chen, C.C., Chen, C.K.: Software-related EMI behavior of embedded microcontroller. In: 2014 IEEE International Symposium on Electromagnetic Compatibility (EMC), pp. 118–122, August 2014
23. Smys, S., Thara Prakash, J., Raj, J.S.: Conducted emission reduction by frequency hopping spread spectrum techniques. Nat. Acad. Sci. Lett. **38**(3), 197–201 (2015)
24. Kwiecień, A., Maćkowski, M., Skoroniak, K.: Reverse engineering of microprocessor program code. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2012. CCIS, vol. 291, pp. 191–197. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31217-5_21
25. Ma, W., Zhao, Z., Meng, J., Pan, Q., Zhang, L.: Precise methods for conducted emi modeling, analysis, and prediction. Sci. China Ser. E Technol. Sci. **51**(6), 641–655 (2008)
26. Yuan, S.Y., Su, W.B., Ho, H.P.: A software technique for EMI optimization, May 2012
27. Kreitlow, M., Garbe, H., Sabath, F.: Influence of software effects on the susceptibility of Ethernet connections. In: 2014 IEEE International Symposium on Electromagnetic Compatibility (EMC), pp. 544–548, August 2014
28. Kwiecień, A., Stój, J.: The cost of redundancy in distributed real-time systems in steady state. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2010. CCIS, vol. 79, pp. 106–120. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13861-4_11

29. Wideł, S., Flak, J., Gaj, P.: Interpretation of dual peak time signal measured in network systems. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2010. CCIS, vol. 79, pp. 141–152. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13861-4_14

The Method of Isochronous Cycle Duration Measurement for Serial Interface IEEE 1394A

Michał Sawicki(✉) and Andrzej Kwiecień

Institute of Informatics, Silesian University of Technology, Gliwice, Poland
{msawicki,akwiecien}@polsl.pl

Abstract. On one bus IEEE 1394A may be a lot of protocols (eg. IIDC and SBP-2) that interact. On this bus cycle jitter may occur, which is not desired in A/V systems. This paper presents a method for measuring isochronous cycle duration. This method allows detection of cycle jitter. It is based on dedicated IEEE 1394 Device Driver and do not require reorganization of a topology of communication system. This article presents the results of measurement of cycle duration in communication system under test.

Keywords: Serial interface · IEEE 1394A · FireWire
Isochronous transfer · Cycle jitter

1 Introduction

A computer system can contain many different devices that communicate via the same interface. Therefore, the protocol designer must take into account the interaction between different communication protocols existing on the same bus. An example of this situation is a vision system based on the IEEE 1394 bus (FireWire), which includes recording devices and external mass storage. Figure 1 shows a system consisting of a FireWire camera, a computer and an external mass storage (hard drive) equipped with a FireWire port. There are two independent transfers executed in this case:

- an isochronous image transfer from camera to computer workstation supervised by IIDC protocol [2],
- an asynchronous data transfer between computer workstation and external mass storage under SBP-2 protocol [3].

It is the communication system with the single FireWire bus over which two communication protocols (IIDC and SPB-2) exist and operate with different types of data transfers (isochronous and asynchronous). The asynchronous transfer may influence the isochronous transfer and disturb the regularity of

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 206–215, 2018.
https://doi.org/10.1007/978-3-319-92459-5_17

isochronous data supply, that is basic requirement imposed on transfer of images. An example of this effect is the oscillation of isochronous cycle duration (called cycle jitter).

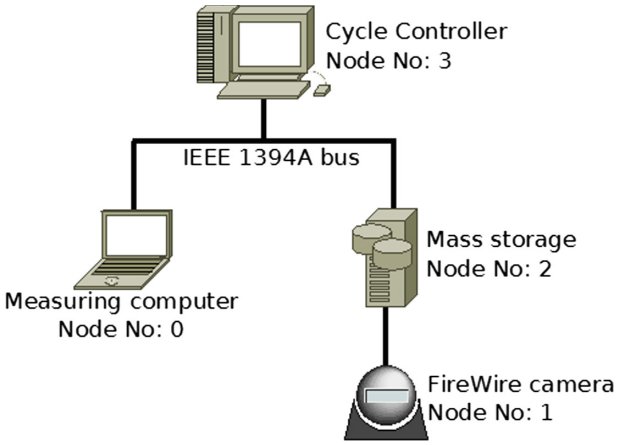


Fig. 1. Scheme of the FireWire communication system

This article presents a method of the cycle jitter detection. It allows to estimate the isochronous cycle duration and detect its oscillation. The first part of this paper contain characterization of isochronous cycle and explanation of cycle jitter. The second part presents a method of measuring the isochronous cycle duration and describes the tools used for this purpose.

Some related works, which presents the analysis of FireWire protocol, can be found in [7].

2 Isochronous Transfer

The IEEE 1394A standard defines two types of data transfer [6]: asynchronous and isochronous. Asynchronous data transfer is used to communicate with mass storage [12] and it is not subject of this article.

Isochronous transfer provides the regularity of data supply (data delivery to a receiver at regular intervals called isochronous cycles). In addition, isochronous transfer provides correctness control of received data, however corrupted data are not retransmitted.

Isochronous transfer is used to deliver short-life data, like video data in the system in Fig. 1. FireWire port allows only isochronous write operation. Operations are assigned to isochronous channels (time slots), in which they are executed [4]. These operations allow to write the same data to one or multiply devices simultaneously.

2.1 Isochronous Cycle

Isochronous data transfer is divided into smaller units (called transactions) executed within the 125 μs isochronous cycles (Fig. 2). Each transaction delivers basic amount of data specified by buffer's BytesPerFrame parameter [8]. At the beginning of each isochronous cycle, the Cycle Master (device located on the top of FireWire tree topology) generates a cycle start packet (CSP). Then, by the isochronous arbitration process nodes are selected to transmit data in successive isochronous channels. After completion of the isochronous transactions, if in the cycle still is free bandwidth [7], the asynchronous data transactions are executed (asynchronous transfer is also divided into transactions).

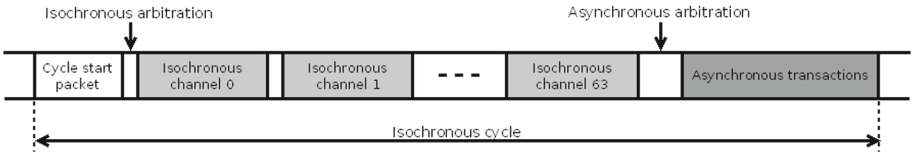


Fig. 2. Isochronous cycle in the FireWire port

2.2 Isochronous Cycle Jitter

In the FireWire interface may occur oscillations of the interval between successive transactions of the same isochronous transfer (Fig. 3). These oscillations are due to the shift of the beginning of the next cycle caused by the last asynchronous transaction extension in the previous cycle.[1] The cycle jitter is undesirable in audio-visual (A/V) systems.

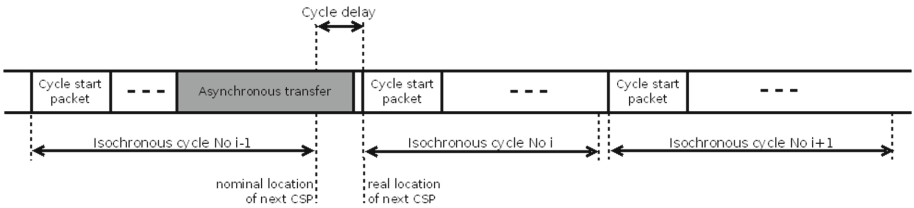


Fig. 3. Isochronous cycle jitter

[1] The requirement on the FireWire bus is not to break but to complete an asychronous transaction, which started before the end of nominal ischronous cycle.

2.3 Isochronous Arbitration

Isochronous arbitration is a FireWire bus access method [9] applyed by nodes that want to send isochronous data and is executed during gaps in the transmission (when bus is at idle). After the CSP broadcast, isochronous nodes, which initiate write isochronous requests, take part in the isochronous arbitration. After completion of the arbitration the node is selected, which can access the bus to execute one isochronous transaction (one isochronous channel).

After completion of the isochronous transaction next nodes compete for bus access. During isochronous arbitration, competing nodes indicate the desire to transmit (generate interface state TX_REQUEST). This signal is passed through successive layers of the communication system topology, up to the root (node located in the first layer). The root grants access the bus for node which TX_REQUEST comes first to one of its ports.

It is obvious that the nodes located closer to the root have precedence in access to the bus. This feature was used in the method described in Sect. 4.1.

3 Software Access to the FireWire Port

Communication between user's application and a device requires a specialized driver, which is installed on operating system. In Microsoft Windows, device driver works in kernel-mode, so the user application (which works in user-mode) does not have programmable access to the device. Therefore, applications need a special diver to communicate with the device. This driver provides an API for the programmer to manage the device.

VHPD1394 device driver (Versatile High Performance IEEE 1394 Device Driver) ensures communication between user application and device via FireWire bus, allowing multiple FireWire devices and multi-user application at the same time.

3.1 Architecture of VHPD1394 Device Driver

VHPD1394 device driver is one of many modules in the FireWire stack on Microsoft Windows (Fig. 4). It cooperates with FireWire bus driver, which provides an interface for FireWire device drivers. On the other hand VHPD1394 provides software interface for user's application.

This software interface allows to perform isochronous transfers (sending data in isochronous channel) as well as receiving data in the indicated isochronous channel. If packets of the same transfer are transmitted in the following cycles, then data is received with frequency 8 kHz. Microsoft Windows is not Real-Time operating system, so VHPD1394 groups received isochronous packets in the streaming buffers. User's application does not process single packet but processes single buffer. This buffer does not contain packets, which have been damaged.

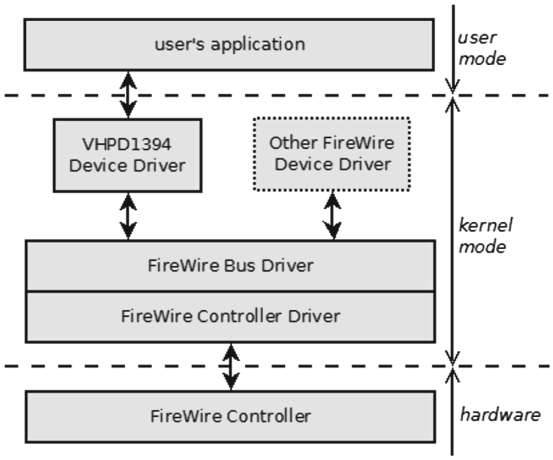


Fig. 4. IEEE 1394 driver stack in Microsoft Windows with VHPD1394

3.2 Data Transmission Using Buffer Queue Streaming Mode

Isochronous data exchange between user's application and VHPD1394 may be performed in the Buffer Queue Streaming Mode. In this mode, the application allocates a buffer pool (Fig. 5), which contains at least two buffers. Received isochronous packets are sent to the currently active buffer. When buffer is full of packets VHPD1394 switches (rotates) buffers: filled buffer is passed to the user's application and another empty buffer from pool is passed to the VHPD1394 driver. The application is notified of rotation of buffers in the pool. Buffer processed by the application is returned to the buffer pool and waits to fill by VHPD1394 driver. In this way it is possible to maintain a continuous flow of isochronous data. In a similar way data may be transmitted in isochronous channel.

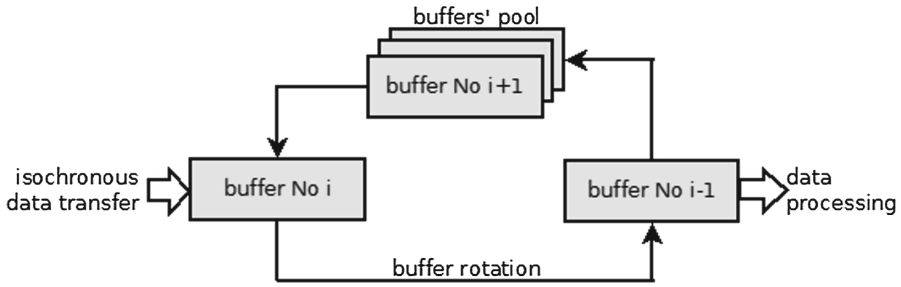


Fig. 5. Buffer Queue Streaming Mode

4 Isochronous Cycle Duration Measurement

To detect cycle jitter, cycle duration must be measured. If the measured time differs from the nominal value (125 μs) then cycle jitter occurred on the FireWire bus.

4.1 The Method of Isochronous Cycle Duration Measurement

Each node has a 32-bit counter Cycle_Time_Register, which is in the address space of the node [1] and is incremented at frequency of 25 MHz. At the beginning of the isochronous cycle Cycle Controller broadcasts CSP packet using asynchronous write broadcast. After receiving CSP packet each node synchronizes its local cycle time (value of Cycle_Time_Register) with time passed as CSP data [5]. Thus nominally every 125 μs each node performs synchronization cycle time with Cycle Controller. Therefore it is not possible to directly read cycle duration on the local node only on the basis of the value of its Cycle_Time_Register.

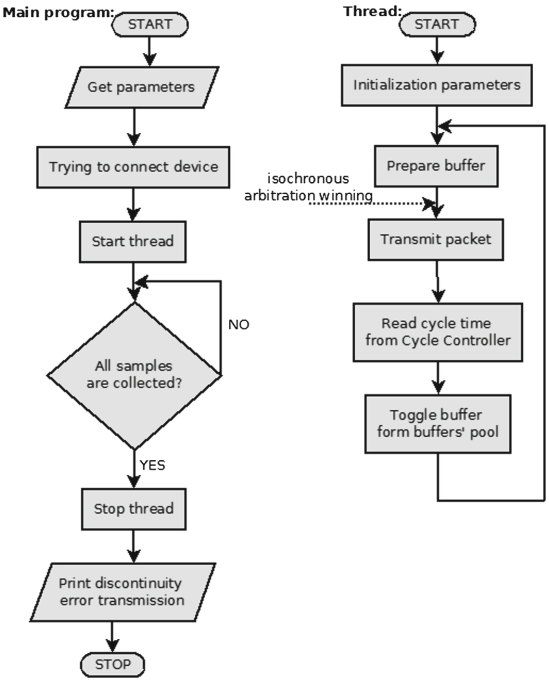


Fig. 6. Block diagram of described method

This problem can be solved by adding the node responsible for measuring the cycle duration to the existing communication system. This node generates

isochronous packets with smallest data field (4 bytes) using Buffer Queue Streaming Mode and simultaneously measures the duration between successive rotations of buffers in the pool. Buffers contain only one isochronous packet. Figure 6 shows described method (as block diagram). This method has been implemented in the sample FireWire communication system (Fig. 1) using multithreading and VHDP1394 Device Driver.

This node should be connected close to the system root (Cycle Controller), because immediately after CSP packet broadcast it must win isochronous arbitration process and get bus access. Then it performs one isochronous transaction and measures a moment of the beginning of cycle (moment of the buffers rotation in the pool). After each two cycles we get cycle duration by subtracting two measured values of cycle time.

4.2 Cycle Time Measurement in the Communication System Under Test

The communication system under test (Fig. 1) consisted of four nodes: one asynchronous (external hard drive), two isochronous (Cycle Controller and FireWire camera) and one node responsible for cycle duration measurement (Measuring Computer).

In the first part of the study cycle duration without asynchronous data transfer between external hard drive and computer was measured. Figure 7 shows the distribution of measured isochronous cycle duration (the first graph). The measured values are focused on the nominal cycle duration (125 μs), what could be expected. This distribution is unimode, that confirms the absence of the cycle jitter.

Microsoft Windows XP Professional was installed on measuring computer. This operating system is not Real-Time system. If user's application does not prepare a buffer before the rotation of buffers in the pool, then transmission discontinuity errors may occur. Therefore, for a sufficiently small number of buffers in the pool these errors may occur. Before the cycle duration measurement, the suitable (minimum) number of buffers in the pool was set, to protect against discontinuity errors. The minimum number of buffers depends on a computer, and may be different for various computers.

In the second part of the study cycle duration with asynchronous data transfer between external hard drive and computer was measured. In this case, cycle jitter may occur. Figure 7 shows the distribution of measured isochronous cycle duration (second graph). This distribution is multimode which confirms the existence of the cycle jitter.

The Fig. 8 shows the cycle duration for two sizes of data field in isochronous packet generated by measuring computer. Increasing size of data field reduces available bandwidth for asynchronous data transfer. Therefore, for larger data packet cycle duration extension is greater than for smaller one. The buffer size was set to 4 bytes to protect FireWire bus against cycle jitter caused by too large buffer in measuring computer (computer generating isochronous packets).

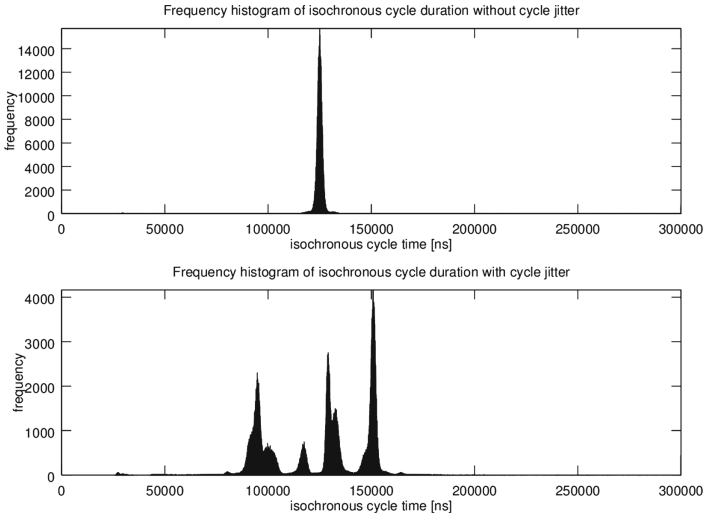


Fig. 7. The histogram of isochronous cycle time

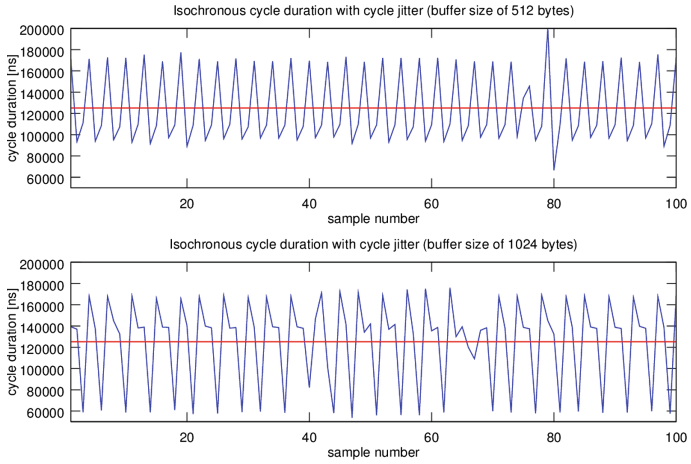


Fig. 8. Isochronous cycle duration with cycle jitter

5 Conclusions

The presented method allows to measure isochronous cycle duration and detect the cycle jitter on the FireWire bus. The cycle jitter is the effect of asynchronous transaction that was not completed within nominal isochronous cycle. Usually it happens, when different communication protocols are active at the same time on the FireWire bus. Cycle jitter is undesirable in communication systems [10] (e.g. A/V system), which use isochronous transfer.

This method uses dedicated IEEE 1394 Device Driver and do not require reorganization of a topology of FireWire bus. This is the advantage of this method, because cycle duration measurement and cycle detection do not require expensive protocol analyzers [7,11]. This method uses the Buffer Queue Streaming Mode, which allows detection of the start of isochronous cycle and register cycle time.

Another way to measure the isochronous cycle (and cycle jitter) is to use an expensive protocol analyzer. However, the method presented in the paper does not require the purchase of FireWire hardware analyzer.

The proposed method has been implemented and tested in a real communication system (Fig. 1). Cycle duration measurement was performed for two cases: without asynchronous transfer and with asynchronous data transfer between a computer and external hard drive, which allowed for cycle jitter detection in communication system under test (Fig. 7).

References

1. IEEE Std 1394A–2000: IEEE Standard for High Performance Serial Bus
2. IIDC2 Digital Camera Control Specification Ver. 1.0.0 (2012)
3. Serial Bus Protocol 2 Specification
4. Anderson, D.: FireWire System Architecture, Mindshare. Inc. Addison-Wesley Developers Press, Boston (2000)
5. Park, S., Jang, I., Lee, S., Choi, S., Cho, K., Lee, J.: Improved cycle time synchronization method for isochronous data transfer on wireless 1394 network. In: The 2008 IEEE International Conference on Ultra-Wideband, Hannover (2008)
6. Zahariadis, T., Pramataris, K.: Multimedia home networks: standards and interfaces. Comput. Stand. Interfaces **24**(5), 425–435 (2002)
7. Steinberg, D., Birk, Y.: An empirical analysis of the IEEE-1394 serial bus protocol. IEEE Micro **20**(1), 58–65 (2000)
8. VHPD1394 Versatile High Performance IEEE 1394 Device Driver for Windows Reference Manual, Ilmenau (2010)
9. Mielczarek, W.: Digital serial bus FireWire. Silesian University of Technology, Gliwice (2010)
10. Domański, A., Domańska, J., Czachórski, T., Klamka, J.: The use of a non-integer order PI controller with an active queue management mechanism. Int. J. Appl. Math. Comput. Sci. **26**(4), 777–789 (2016)

11. Maćkowski, M.: The influence of electromagnetic disturbances on data transmission in USB standard. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2009. CCIS, vol. 39, pp. 95–102. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02671-3_11
12. Sawicki, M.: Analysis of asynchronous data transfer in communication system models for USB and FireWire. In: Scientific Conference Computer Networks 2012, Studia Informatica, vol. 33, no 1A (107), Szczyrk (2012)

Queueing Theory

Time to Buffer Overflow in a Queueing Model with Working Vacation Policy

Wojciech M. Kempa(✉) and Martyna Kobielnik

Faculty of Applied Mathematics, Silesian University of Technology,
23 Kaszubska Street, 44-100 Gliwice, Poland
{wojciech.kempa,martyna.kobielnik}@polsl.pl

Abstract. A finite-buffer queueing model with Poisson arrivals and exponential processing times is investigated. Every time when the system becomes empty, the service station begins a generally distributed single working vacation period, during which the processing is provided with another (slower) rate. After the completion of the vacation period the processing is being continued normally, with original speed. The next working vacation period is being initialized at the next time at which the system becomes empty, and so on. Identifying Markov epochs in the evolution of the considered model, the system of Volterra-type integral equations for the time to buffer overflow tail distribution function, conditioned by the initial buffer state, is built. The solution of the corresponding system written for Laplace transforms is given in a compact-form using the linear algebraic approach. The considered queueing model can be used in modelling of the network node in which the typical processing rate changes periodically, due to e.g. introducing a priority traffic.

Keywords: Buffer overflow · Memoryless property
Total probability law · Transient analysis · Working vacation policy

1 Introduction

Queueing models with different-type vacation policies are investigated extensively and the number of publications in this topic is still increasing. This kind of systems can be successfully applied in modelling and analyzing the phenomena occurring in telecommunication and computer networks. In particular, they are very useful tool when the service stations (e.g. servers, network nodes) can use the idle period to perform some additional tasks. Different variations of vacation policy can be used then to describe the considered problem more precisely.

One of such variations is the working vacation (WV) policy proposed in 2002 by Servi and Finn. In [10] the authors introduce an $M/M/1$ queue with WV policy to model the behaviour of a reconfigurable WDM (wavelength-division multiplexing) optical access network. Each time the server with WV finishes the processing of the last job waiting in the queue, it switches to a WV mode. During the WV period another (usually slower) rate of service is offered, instead

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 219–231, 2018.
https://doi.org/10.1007/978-3-319-92459-5_18

of suspending it completely. The idle resources can be used simultaneously to accomplish tasks not related to the processing of the queue, e.g. some maintenance work. Some variants of WV systems can be applied e.g. in modelling of the operation of a network node in which the typical processing rate changes periodically, due to e.g. introducing a priority traffic.

Baba [1] extended the investigation of WV queues to a $GI/M/1$ queue, and derived the steady-state distribution for the number of customers in the system. In [14] Wu and Takagi obtained the stationary distributions for the queue size and the time a customer stays in the system in case of an $M/G/1$ WV model. Banik [2] used the method of embedded Markov chain and a supplementary variable to compute the queue-length distribution, the probability of blocking, and the mean waiting time in the $GI/M/1$ WV queue with finite buffer. An $M/G/1$ retrial queue was analyzed in [4]. The method of supplementary variable was used to obtain the stationary queue-size distribution in the case of a model with vacation interruption. In [12] the authors investigated an $M/M/1$ retrial queue with WV and feedback under N-policy. The probability generating function of the number of customers present in the system in the steady state of an $M/G/1$ queue with exponential WV and gated service discipline was obtained in [9].

While the literature concerning WV queues is growing fast, most of the results relates only to the stationary characteristics of the system. However, in some cases the steady-state analysis does not give the sufficient insight into the system's behaviour, e.g. in some situations the convergence rate to the stationary distributions can be very slow. The transient analysis can also be imposed by the high variability of the system parameters or the need of observing the behaviour of the server shortly after its opening or applying a new control mechanism.

The transient results for queueing models with the WV regime can be found e.g. in [11,13]. In [11] the time-dependent system-size probabilities were obtained in the case of the $M/M/1$ queue with heterogeneous service and customers' impatience. One can find the transient distribution of the number of customers in the case of the $M/M/c$ model in [13]. In [6,7] an $M/M/1/N$ queue with WV policy was investigated and transient distributions of the queue size and the virtual waiting time of a customer were obtained. In [5] the time dependent analysis of queueing delay behaviour in the case of a $GI/M/1/N$ queue was accomplished.

This paper concerns the analysis of an $M/M/1/N$ queue with a generally distributed working vacations. The incoming stream of packets is described by a Poisson process with rate λ. When, after completion of processing a job in normal mode (with rate μ), the server finds the queue empty, it switches to the WV mode and it comes back to the normal mode after a random, generally distributed, amount of time with the cumulative distribution function (CDF) $G(\cdot)$. During the WV period, the arriving jobs are served with the different rate μ^* ($<\mu$). After completion of the WV, the rate of service is switched back to μ. If a job enters the system while there are already N jobs present, it is lost due to the buffer overflow. In the paper we deal with a CDF of the time to the first buffer overflow in the considered model. Applying the analytical approach

based on identifying Markov moments in the evolution of the system, continuous total probability formula and linear algebra, we obtain the representation for the Laplace transform (LT) of the tail CDF of the time to the first buffer overflow, conditioned by the number of packets accumulated in the buffer at the starting moment.

The phenomenon of losses is typical in packet-oriented telecommunication and computer networks (e.g. in IP routers) or in automated production systems, where some tasks have to be redirected to another production line or a magazine in the case of exhaustion of the available capacity for accumulation the incoming traffic. As it seems, the well-known performance measure as the loss ratio does not give sufficient knowledge about the probabilistic nature of the process of losses. The in-depth analysis requires e.g. the information about the probability distribution of the time to buffer saturation.

The rest of the paper is organized as follows. In Sect. 2 the investigated problem is defined, and a useful algebraic result is presented. Sections 3 and 4 contain the necessary solutions used in Sect. 5 to solve the main problem. In Sect. 5 the LT of the time to buffer overflow distribution is derived using algebraic approach based on theorem stated in Sect. 2. Section 6 contains some numerical examples.

2 Equations for Time-to-Buffer-Overflow Distribution

Introduce the CDF of the time τ to the first buffer overflow as follows:

$$B_n(t) \stackrel{def}{=} \mathbf{P}\left\{\tau > t \,|\, X(0) = n\right\}, \quad t > 0, \tag{1}$$

where $X(0)$ stands for the initial buffer state, $n \in \{0, 1, ..., N-1\}$.

We assume, that if the queue is empty at the opening, then the WV period initialize the evolution of the system. Otherwise, it starts the operation in normal mode. Using the method of embedded Markov chain and the formula of total probability with respect to the moment when WV period ends, we can write the first equation (for $n = 0$):

$$B_0(t) = \int_0^t \sum_{k=0}^{N-1} P_0^{slow}(u,k) B_k(t-u) \mathrm{d}G(u) + [1 - G(t)] B_0^{slow}(t), \tag{2}$$

where

$$P_0^{slow}(u,k) = \mathbf{P}\left\{ \left(X(u) = k\right) \wedge \left(\sup_{v \in [0,u]} X(v) \leq N-1 \right) | \, X(0) = 0 \right\} \tag{3}$$

is the probability that in the time interval of length u the number of jobs present in the system will change from 0 to k, and it will not reach N (i.e. the buffer overflow will not occur) and

$$B_0^{slow}(t) \stackrel{def}{=} \mathbf{P}\left\{\tau > t \,|\, X(0) = 0\right\}, \quad t > 0$$

is the CDF of the time to the first buffer overflow in the $M/M/1/N$ system without WV policy where the parameter of input stream is the same as in original model and the service parameter is μ^*. In the case the system starts the evolution being non-empty and in normal mode, using the formula of total probability with respect to the first Markov event after the opening, we get, similarly

$$B_n(t) = \lambda \int_0^t \mathrm{e}^{-(\lambda+\mu)x} B_{n+1}(t-x)\mathrm{d}x + \mu \int_0^t \mathrm{e}^{-(\lambda+\mu)x} B_{n-1}(t-x)\mathrm{d}x + \mathrm{e}^{-(\lambda+\mu)t}, \quad (4)$$

where $1 \leq n \leq N-2$, and, finally,

$$B_{N-1}(t) = \mu \int_0^t \mathrm{e}^{-(\lambda+\mu)x} B_{N-2}(t-x)\mathrm{d}x + \mathrm{e}^{-(\lambda+\mu)t}. \quad (5)$$

In (4)–(5) the summand with coefficient λ (μ) in front of the integral corresponds to the situation, where the arrival of a new job (the end of processing a job) occurs first. The last summand is the probability of no events before t.

To find explicit formulae for the functions $B_n(t)$, $0 \leq n \leq N-1$, we need to find the representations of $P_0^{slow}(t,k)$ and $B_0^{slow}(t)$. In the analytical approach we use the following algebraic result (see [8]):

Theorem 1. *Let (τ_k), $k \geq 0$, $\tau_0 \neq 0$ and (θ_k), $k \geq 1$ be two given sequences. Each solution of the following system of linear equations:*

$$\sum_{k=-1}^{n-2} \tau_{k+1} y_{n-k} - y_n = \theta_n, \ n \geq 2,$$

can be written in the following way:

$$y_n = M R_{n-1} + \sum_{k=2}^{n} R_{n-k}\theta_k, \ n \geq 2,$$

where $M \in \mathbb{R}$, and (R_k), $k \geq 0$ is a sequence defined recursively as follows:

$$R_0 = 0, \ R_1 = \tau_0^{-1}, \ R_{k+1} = R_1(R_k - \sum_{i=0}^{k} \tau_{i+1} R_{k-i}).$$

3 Time to Buffer Overflow in a Reliable System

Let us consider a reliable system (without working vacation policy) that processes the arriving packets with a slower speed μ^*. We can write the following system of equations (using the formula of total probability with respect to the first event after opening):

$$B_0^{slow}(t) = \lambda \int_0^t \mathrm{e}^{-\lambda x} B_1^{slow}(t-x)\mathrm{d}x + \mathrm{e}^{-\lambda t},$$
$$B_n^{slow}(t) = \lambda \int_0^t \mathrm{e}^{-(\lambda+\mu^*)x} B_{n+1}^{slow}(t-x)\mathrm{d}x + \mu^* \int_0^t \mathrm{e}^{-(\lambda+\mu^*)x} B_{n-1}^{slow}(t-x)\mathrm{d}x + \mathrm{e}^{-(\lambda+\mu^*)t}, \tag{6}$$

where $1 \leq n \leq N-2$, and

$$B_{N-1}^{slow}(t) = \mu^* \int_0^t \mathrm{e}^{-(\lambda+\mu^*)x} B_{N-2}^{slow}(t-x)\mathrm{d}x + \mathrm{e}^{-(\lambda+\mu^*)t}. \tag{7}$$

The interpretation of this equations is similar to the analogous ones from Sect. 2. After introducing LTs $b_i^{slow}(s) = \int_0^\infty \mathrm{e}^{-st} B_i^{slow}(t)\mathrm{d}t$, $0 \leq i \leq N-1$, we can rewrite this system in the form

$$b_0^{slow}(s) = \frac{\lambda}{\lambda+s} b_1^{slow}(s) + \frac{1}{\lambda+s}, \tag{8}$$
$$b_n^{slow}(s) = \frac{\lambda}{\lambda+\mu^*+s} b_{n+1}^{slow}(s) + \frac{\mu^*}{\lambda+\mu^*+s} b_{n-1}^{slow}(s) + \frac{1}{\lambda+\mu^*+s}, \tag{9}$$

where $1 \leq n \leq N-2$, and

$$b_{N-1}^{slow}(s) = \frac{\mu^*}{\lambda+\mu^*+s} b_{N-2}^{slow}(s) + \frac{1}{\lambda+\mu^*+s}. \tag{10}$$

If we define the following sequence:

$$a_0^*(s) = \frac{\lambda}{\lambda+\mu^*+s}, \quad a_2^*(s) = \frac{\mu^*}{\lambda+\mu^*+s}, \quad a_k^*(s) = 0,\ k \neq 0, 2,$$

and

$$\phi_n^*(s) = -\frac{1}{\lambda+\mu^*+s} - \delta_{n,1} a_2^*(s) b_0^{slow}(s),$$

where $\delta_{i,j}$ is the Kronecker delta function (i.e. $\delta_{i,j} = 1$ when $i = j$ and $\delta_{i,j} = 0$ otherwise), then the system (9) can be rewritten as follows:

$$\sum_{k=-1}^{n-1} a_{k+1}^*(s) b_{n-k}^{slow}(s) - b_n^{slow}(s) = \phi_n^*(s), \quad 1 \leq n \leq N-2. \tag{11}$$

In consequence, the solution of (8), (10) and (11) can be written in the following way (see Theorem 1):

$$b_n^{slow}(s) = M^*(s) R_n^*(s) + \sum_{k=1}^{n} R_{n-k}^*(s) \phi_k^*(s), \quad 1 \leq n \leq N-1, \tag{12}$$

where $M^*(s)$ is an unknown function, and $R_k^*(s)$ is a sequence defined recursively as follows:

$$R_0^*(s)=0,\ R_1^*(s)=\left(a_0^*(s)\right)^{-1},\ R_{k+1}^*(s)=R_1^*(s)\left(R_k^*(s)-a_2^*(s)R_{k-1}^*(s)\right), \tag{13}$$

for $1\le k\le N-2$.

Introducing (12) into the right side of (10), we get:

$$\begin{aligned} b_{N-1}^{slow}(s)=&a_2^*(s)\Big[M^*(s)R_{N-2}^*(s)+\sum_{k=1}^{N-2}R_{N-2-k}^*(s)\\ &\times\Big(-\frac{1}{\lambda+\mu^*+s}-\delta_{1,k}a_2^*(s)b_0^{slow}(s)\Big)\Big]+\frac{1}{\lambda+\mu^*+s}. \end{aligned} \tag{14}$$

Similarly, writing (12) for $n=N-1$, we obtain:

$$b_{N-1}^{slow}(s)=M^*(s)R_{N-1}^*(s)+\sum_{k=1}^{N-1}R_{N-1-k}^*(s)\Big(-\frac{1}{\lambda+\mu^*+s}-\delta_{1,k}a_2^*(s)b_0^{slow}(s)\Big)\Big]. \tag{15}$$

Comparing the right sides of (14) and (15), we eliminate $M^*(s)$ as a function of $b_0^{slow}(s)$:

$$\begin{aligned} M^*(s)=&\left(\lambda+\mu^*+s\right)^{-1}\left(a_2^*(s)R_{N-2}^*(s)-R_{N-1}^*(s)\right)^{-1}\\ &\times\Big[\sum_{k=1}^{N-2}\left(a_2^*(s)R_{N-k-2}^*(s)-R_{N-k-1}^*(s)\right)+\mu^*b_0^{slow}(s)\left(a_2^*(s)R_{N-3}^*(s)-R_{N-2}^*(s)\right)-1\Big]. \end{aligned} \tag{16}$$

Referring now to (8), (12) and (16), we get

$$\begin{aligned} b_0^*(s)=&\frac{1}{\lambda+s}\Bigg\{\lambda R_1^*(s)\Big(a_2^*(s)R_{N-2}^*(s)-R_{N-1}^*(s)\Big)^{-1}\\ &\times\Bigg[\sum_{k=1}^{N-2}\Big(-\frac{1}{\lambda+\mu^*+s}-\delta_{1,k}a_2^*(s)b_0^{slow}(s)\Big)\left(R_{N-1-k}^*(s)-a_2^*(s)R_{N-2-k}^*(s)\right)\\ &-\frac{1}{\lambda+\mu^*+s}\Bigg]+1\Bigg\}, \end{aligned}$$

and hence, after simplification,

$$b_0^{slow}(s)=T_1^{slow}(s)T_2^{slow}(s), \tag{17}$$

where

$$T_1^{slow}(s)\overset{def}{=}\left(s+\lambda+\mu^*\left(a_2^*(s)R_{N-2}^*(s)-R_{N-1}^*(s)\right)^{-1}\left(a_2^*(s)R_{N-3}^*(s)-R_{N-2}^*(s)\right)\right)^{-1} \tag{18}$$

and

$$T_2^{slow}(s)\overset{def}{=}\frac{\left(\sum_{k=0}^{N-2}\left(a_2^*(s)R_{N-k-2}^*(s)-R_{N-k-1}^*(s)\right)-1\right)}{\left(a_2^*(s)R_{N-2}^*(s)-R_{N-1}^*(s)\right)}. \tag{19}$$

4 Solution for $P_0^{slow}(t,m)$

Let us note that if we take (see (3))

$$P_n^{slow}(t,m) = \mathbf{P}\left\{\left(X(t)=m\right) \wedge \left(\sup_{v\in[0,t]} X(v) \le N-1\right) \mid X(0)=n\right\},$$

where $0 \le n \le N-1$, then the following system is satisfied:

$$\begin{aligned} P_0^{slow}(t,m) &= \lambda \int_0^t \mathrm{e}^{-\lambda x} P_1^{slow}(t-x,m)\mathrm{d}x + \delta_{m,0}\mathrm{e}^{-\lambda t}, \\ P_n^{slow}(t,m) &= \lambda \int_0^t \mathrm{e}^{-(\lambda+\mu^*)x} P_{n+1,m}^{slow}(t-x,m)\mathrm{d}x \\ &\quad + \mu^* \int_0^t \mathrm{e}^{-(\lambda+\mu^*)x} P_{n-1}^{slow}(t-x,m)\mathrm{d}x + \delta_{m,n}\mathrm{e}^{-(\lambda+\mu^*)t}, \end{aligned}$$

where $1 \le n \le N-2$, and

$$P_{N-1}^{slow}(t,m) = \mu^* \int_0^t \mathrm{e}^{-(\lambda+\mu^*)x} P_{N-2}^{slow}(t-x,m)\mathrm{d}x + \delta_{m,N-1}\mathrm{e}^{-(\lambda+\mu^*)t}.$$

After introducing LTs $p_i^{slow}(s) = \int_0^\infty \mathrm{e}^{-st} P_i^{slow}(t)\mathrm{d}t$, $0 \le i \le N-1$, we get

$$p_0^{slow}(s,m) = \frac{\lambda}{\lambda+s} p_1^{slow}(s,m) + \frac{\delta_{m,0}}{\lambda+s}, \tag{20}$$

$$p_n^{slow}(s,m) = \frac{\lambda}{\lambda+\mu^*+s} p_{n+1}^{slow}(s,m) + \frac{\mu^*}{\lambda+\mu^*+s} p_{n-1}^{slow}(s,m) + \frac{\delta_{m,n}}{\lambda+\mu^*+s}, \tag{21}$$

where $1 \le n \le N-2$, and

$$p_{N-1}^{slow}(s,m) = \frac{\mu^*}{\lambda+\mu^*+s} p_{N-2}^{slow}(s,m) + \frac{\delta_{m,N-1}}{\lambda+\mu^*+s}. \tag{22}$$

Comparing (20)–(22) to (8)–(10), we conclude that (compare (17)–(19))

$$p_0^{slow}(s,m) = T_1^{slow}(s) T_3^{slow}(s,m),$$

where

$$T_3^{slow}(s,m) \stackrel{def}{=} \frac{\left(\sum_{k=0}^{N-2} \delta_{m,k}\left(a_2^*(s) R_{N-k-2}^*(s) - R_{N-k-1}^*(s)\right) - \delta_{m,N-1}\right)}{\left(a_2^*(s) R_{N-2}^*(s) - R_{N-1}^*(s)\right)}.$$

5 Main Result

Introducing LTs $b_i(s) = \int_0^\infty \mathrm{e}^{-st} B_i(t)\mathrm{d}t,\ 0 \le i \le N-1$ into Eqs. (2), (4) and (5), we get

$$b_0(s) = \sum_{k=0}^{N-1} b_k(s)\widehat{p}_k(s) + d(s), \tag{23}$$

$$b_n(s) = \frac{\lambda}{\lambda+\mu+s} b_{n+1}(s) + \frac{\mu}{\lambda+\mu+s} b_{n-1}(s) + \frac{1}{\lambda+\mu+s}, \tag{24}$$

for $1 \le n \le N-2$, and

$$b_{N-1}(s) = \frac{\mu}{\lambda+\mu+s} b_{N-2}(s) + \frac{1}{\lambda+\mu+s}, \tag{25}$$

where we denote

$$\widehat{p}_k(s) \stackrel{def}{=} \int_0^\infty \mathrm{e}^{-st} P_0^{slow}(t,k)\mathrm{d}G(t),$$
$$d(s) \stackrel{def}{=} \int_0^\infty \mathrm{e}^{-st} B_0^{slow}(t)[1-G(t)]\mathrm{d}t.$$

The solution of (24)–(25) is given by (compare (12))

$$b_n(s) = M(s)R_n(s) + \sum_{k=1}^{n} R_{n-k}(s)\phi_k(s), \quad 1 \le n \le N-1, \tag{26}$$

where now the sequence $\big(R_k(s)\big)$ is recursively defined using

$$a_0(s) = \frac{\lambda}{\lambda+\mu+s},\ \ a_2(s) = \frac{\mu}{\lambda+\mu+s},\ \ a_k(s) = 0,\ k \ne 0, 2,$$

and

$$\phi_n(s) = -\frac{1}{\lambda+\mu+s} - \delta_{n,1} a_2(s) b_0(s).$$

Substituting (26) into (23), we obtain

$$b_0(s) = \sum_{k=1}^{N-1} \widehat{p}_k(s)\Big[M(s)R_k(s) + \sum_{i=1}^{k} R_{k-i}(s)\Big(-\frac{1}{\lambda+\mu+s} - \delta_{1,i} a_2(s) b_0(s)\Big)\Big] + b_0(s)\widehat{p}_0(s)b_0(s), \tag{27}$$

and hence

$$M(s) = A(s)b_0(s) + B(s),$$

where

$$A(s) \stackrel{def}{=} \frac{1 + a_2(s)\sum_{k=1}^{N-1} \widehat{p}_k(s) R_{k-1}(s) - \widehat{p}_0(s)}{\sum_{k=1}^{N-1} \widehat{p}_k(s) R_k(s)},$$

and

$$B(s) \stackrel{def}{=} \frac{(\lambda + \mu + s)^{-1} \sum_{k=1}^{N-1} \widehat{p}_k(s) \sum_{i=1}^{k} R_{k-i}(s) - d(s)}{\sum_{k=1}^{N-1} \widehat{p}_k(s) R_k(s)}.$$

The remaining task is to find the representation for $b_0(s)$. To do it, let us use (26) on the right side of (25) and compare to the right side of (26) written for $n = N - 2$. We obtain:

$$b_0(s) = [A(s)\,(a_2(s)R_{N-2}(s) - R_{N-1}(s)) - a_2(s)\,(a_2(s)R_{N-3}(s) - R_{N-2}(s))]^{-1} \tag{28}$$
$$\times \left[\frac{\left(\sum_{k=1}^{N-2} (a_2(s)R_{N-k-2}(s) - R_{N-k-1}(s)) - 1\right)}{(s + \lambda + \mu)} - B(s)(a_2(s)R_{N-2}(s) - R_{N-1}(s)) \right].$$

6 Numerical Examples

In this section, we present some numerical examples for $M/M/1/10$ system with WV. The length of the WV period follows the distribution $G(t) = \frac{1}{2}(1 - \exp(-ct)) + \frac{1}{4}(2 - \exp(-3t) - \exp(-t))$, and the parameters of arrival process, service in normal and in the WV mode are λ, μ and μ^*, respectively. In the following examples, we present the values of $B_0(5)$ in dependence of μ and the mean length of the WV period $\mathbf{E}(T_{WV})$.

Example 1. *Let us consider the $M/M/1/10$ model with WV with $\lambda = 1, 3, 5$, $\mu^* = 1$ and $c = 0.1$ ($\mathbf{E}(T_{WV}) = 0.46$). We compute $B_0(5)$ for ten different values of μ.*

In Figs. 1, 2 and 3 the behaviour of $B_0(5)$ in dependence of μ is presented for $\lambda = 1, 3, 5$. The Fig. 1 shows the numerical result for $\lambda = 1$. We see, that as the value of service rate rises, the probability, that the time to the first buffer overflow is greater than 5 decreases. As one can note, in this scenario the ratio between λ and μ never exceeds 1, therefore the system will cope with arriving jobs and will often take the vacations. In WV mode, the server operates with rate $\mu^* = 1$ which is equal to λ, thus the more time the system spends in WV mode, the probability of buffer overflow before the moment t increases.

As shown on the Figs. 2 and 3, $B_0(5)$ increases as long as the ratio $\rho = \lambda/\mu$ is greater than 1. Then, as described previously, the server will more often take the vacations, thus the probability of no buffer saturation before $t = 5$ decreases.

Example 2. *We consider the $M/M/1/10$ model with WV with $\lambda = 2, 5, 7$, $\mu = 5$ and $\mu^* = 1$. We compute $B_0(5)$ for different values of $\mathbf{E}(T_{WV})$.*

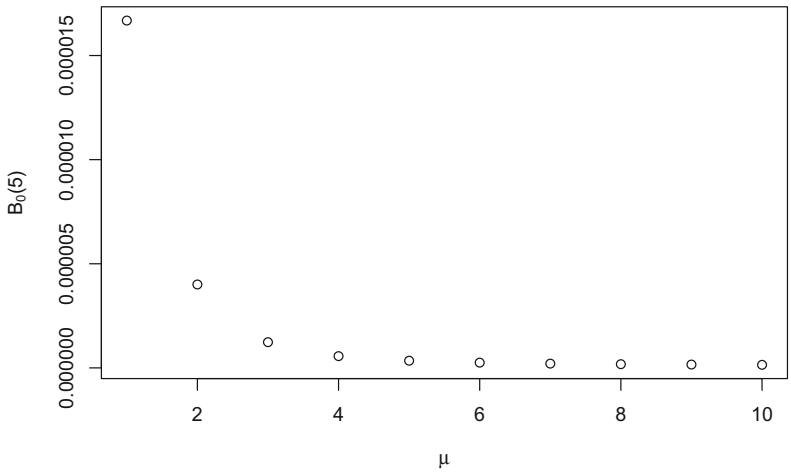


Fig. 1. The values of $B_0(5)$ in dependence of μ for $\lambda = 1$

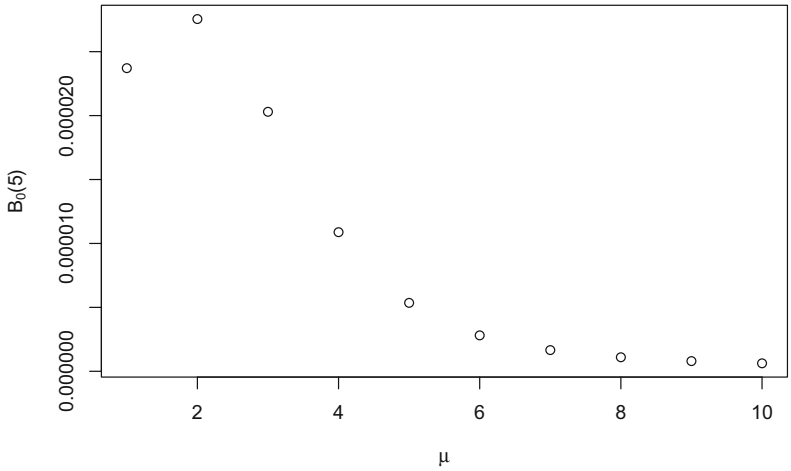


Fig. 2. The values of $B_0(5)$ in dependence of μ for $\lambda = 3$

The Figs. 4, 5 and 6 present values of $B_0(5)$ in dependence of $\mathbf{E}(T_{WV})$ for three different values of λ.

In the first scenario (Fig. 4), when $\lambda = 2$, the probability, that there will be no buffer overflow before $t = 5$ increases simultaneously with the mean length of vacation period. The ratio $\rho = 0.4$ and $\rho^* = \lambda/\mu^* = 1$, therefore, the load of the server never exceedes 1.

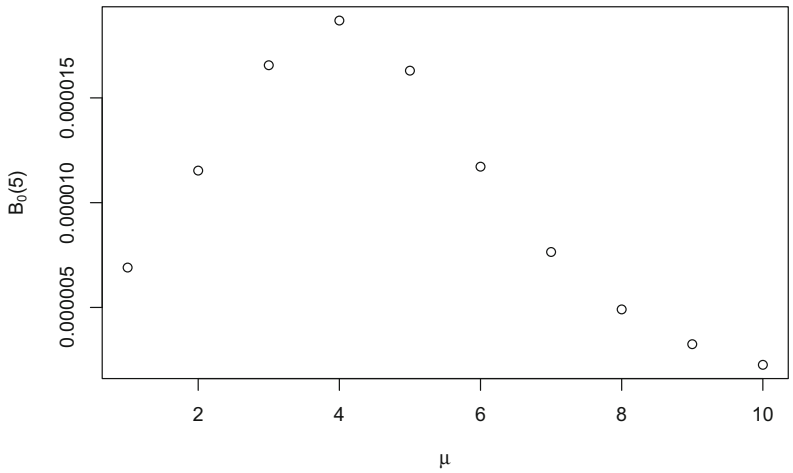


Fig. 3. The values of $B_0(5)$ in dependence of μ for $\lambda = 5$

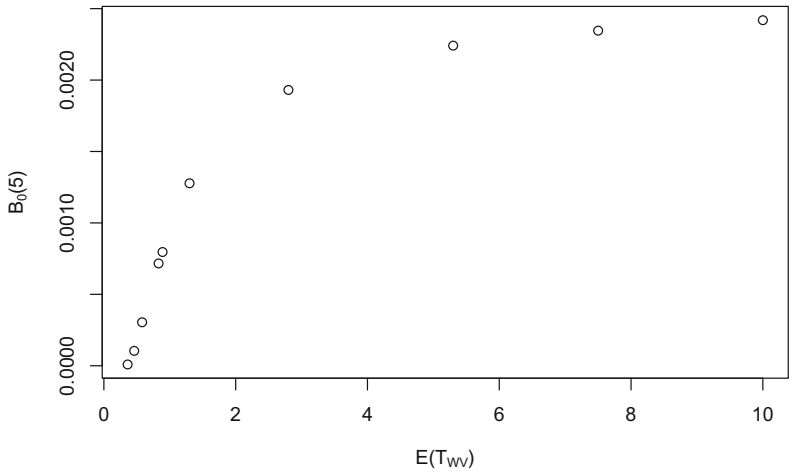


Fig. 4. The values of $B_0(5)$ in dependence of $\mathbf{E}(T_{WV})$ for $\lambda = 2, \mu = 5$ and $\mu^* = 1$.

As one can note, for $\lambda = 5$ (Fig. 5), the probabilities $B_0(5)$ increase in case of short WV period and begin to decrease starting at a value of $\mathbf{E}(T_{WV}) \approx 1$. The longer the WV period lasts, the longer the load is $\rho^* = 5$ instead of $\rho = 1$. The similar observation can be made in the case of $\lambda = 7$ (Fig. 6).

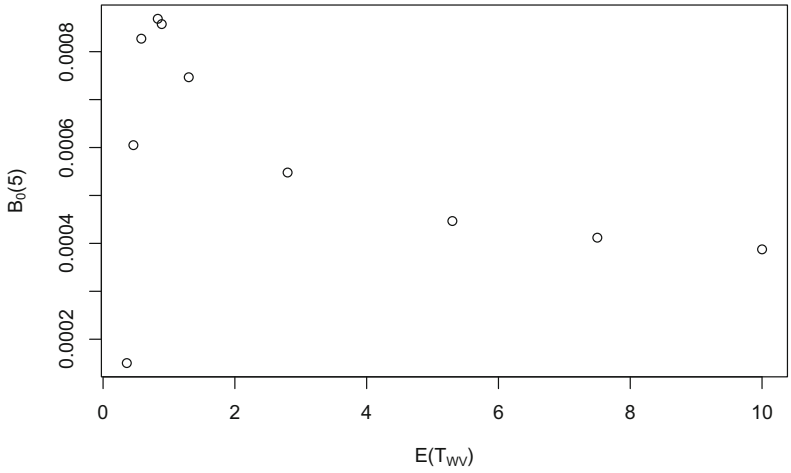


Fig. 5. The values of $B_0(5)$ in dependence of $\mathbf{E}(T_{WV})$ for $\lambda = 5, \mu = 5$ and $\mu^* = 1$.

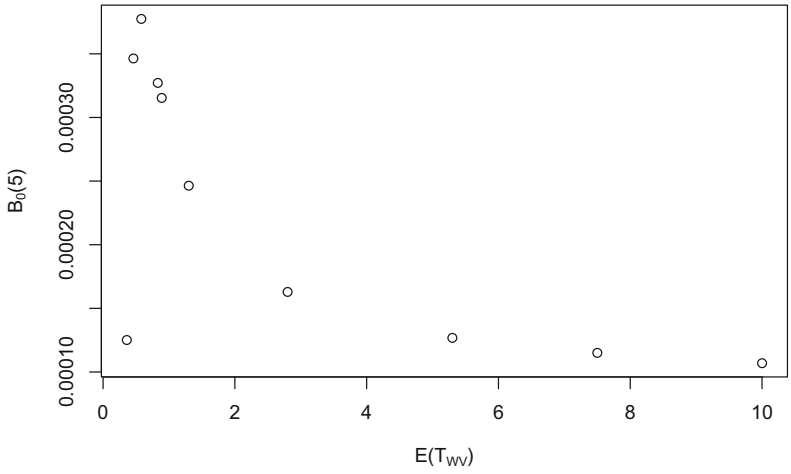


Fig. 6. The values of $B_0(5)$ in dependence of $\mathbf{E}(T_{WV})$ for $\lambda = 10, \mu = 5$ and $\mu^* = 1$.

7 Conclusions

In the paper, the $M/M/1/N-$queue was investigated using the embedded Markov chain technique, the formula of total probability and linear algebra. The explicit representation of the LT of the transient conditional tail CDF of the time to the first buffer overflow is found. To obtain the CDF, one of many different approaches of numerical LT inversion may be applied, e.g. Dubner-Abate or Gaver-Stehfest method (see [3]). To give an insight into the considered model, the obtained result was illustrated with some numerical examples.

References

1. Baba, Y.: Analysis of a GI/M/1 queue with multiple working vacations. Oper. Res. Lett. **33**(2), 201–209 (2005)
2. Banik, A.: Analysis of single working vacation in GI/M/1/N and GI/M/1/∞ queueing systems. Int. J. Oper. Res. (IJOR) **7**(3), 314–333 (2010)
3. Davies, B., Martin, B.: Numerical inversion of the Laplace transform: a survey and comparison of methods. J. Comput. Phys. **33**(1), 1–32 (1979)
4. Gao S., Wang J., Li W. W: An M/G/1 retrial queue with general retrial times, working vacations and vacation interruption. APJOR **31**(2) (2014)
5. Kempa W. M.: Non-stationary analysis of queueing delay behavior in the GI/M/1/N-type queue with server working vacations. In: AIP Conference Proceedings, vol. 1690(1) (2015)
6. Kempa, W.M., Kobielnik, M.: Transient solution for queue-size distribution in a certain finite-buffer model with server working vacations. In: Dregvaite, G., Damasevicius, R. (eds.) ICIST 2016. CCIS, vol. 639, pp. 426–440. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46254-7_34
7. Kempa, W.M., Kobielnik, M.: Transient virtual waiting time distribution in M/M/1/N system with working vacations. In: Proceedings of the 11th Scientific Conference Internet in the Information Society 2016, pp. 305–313 (2016)
8. Korolyuk, V.: Boundary-Value Problems for Compound Poisson Processes. Naukova Dumka, Kiev (1975). (in Russian)
9. Saffer, Z., Telek, M.: $M/G/1$ queue with exponential working vacation and gated service. In: Al-Begain, K., Balsamo, S., Fiems, D., Marin, A. (eds.) ASMTA 2011. LNCS, vol. 6751, pp. 28–42. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21713-5_3
10. Servi, L.D., Finn, S.G.: M/M/1 queues with working vacations (M/M/1/WV). Perform. Eval. **50**(1), 41–52 (2002)
11. Sudhesh, R., Azhagappan, A., Dharmaraja, S.: Transient analysis of M/M/1 queue with working vacation, heterogeneous service and customers' impatience. RAIRO Oper. Res. **51**(3), 591–606 (2017)
12. Tao L., Zhang L., Gao S.: M/M/1 retrial queue with working vacation interruption and feedback under n-policy. J. Appl. Math. **2014**, 414739:1–414739:9 (2014)
13. Vijayashree, K.V., Janani, B.: Transient analysis of an M/M/c queue subject to multiple exponential working vacation. Appl. Math. Sci. **9**(74), 3669–3677 (2015)
14. Wu, D., Takagi, H.: M/G/1 queue with multiple working vacations. Perform. Eval. **63**(7), 654–681 (2006)

Adaptation of the N-GREEN Architecture for a Bursty Traffic

Tulin Atmaca, Amira Kamli(✉), and Artur Rataj

Institut Mines-Télécom, Télécom Sud Paris, Evry, France
{tulin.atmaca,amira.kamli}@telecom-sudparis.eu,
arturrataj@gmail.com

Abstract. N-GREEN is a cost attractive optical ring network which uses coloured packets. It is normally fit to a predictable traffic with a low burst rate, found e.g. in the metro aggregation. Here we try to adapt the network to other, potentially interesting applications where the traffic is more bursty, by proposing a packet management scheme with adaptive expiration times, determined in response to local and/or global queue sizes. The exact relation is found using a direct optimisation method which uses a simulation. We show that thanks to the regulation of the expiration time, an N-GREEN ring may continuously adapt to a bursty/unpredictable traffic of a varying average load, provided the nodes inform one another about the momentary size of data in their input buffers. The adaptation may considerably decrease the latency of the network.

Keywords: Optical network · Coloured packets · N-GREEN
Bursty traffic · Optimisation

1 Introduction

The technology of switching and transmission of optical coloured packets attempts to fulfil the need for networks of extremely high speeds. On the other hand, such optical interfaces are very expensive, which limits their applications. This is why the N-GREEN metro-aggregation network [1] is proposed. It provides streamlined, cost-optimised interfaces [2,3] which allow a transmission at 100 Gbit using 10 fixed wavelengths along a ring, yet without the expensive add-ons like accordable lasers or a possibility of an independent assignment or routing of different wavelengths. In applications like the backhaul metro networks or G5, where the traffic is largely predictable due to a statistical averaging of a large number of transmissions, N-GREEN, as an overdimensioned network in that case, can be even more simplified by removing any substantial buffering and thus queue management from the nodes. This is possible for example if there are up to 10 Ethernet cards, of 10 Gbit each, connected to a single N-GREEN ring.

However, due to its reduced initial, maintenance and energy costs, N-GREEN seems to be attractive also in a number of other applications requiring extreme

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 232–244, 2018.
https://doi.org/10.1007/978-3-319-92459-5_19

transmission rates of data which has a substantially diverse (e.g. bursty) traffic profile, like big data centres or commercial centres. If there is no requirement for a hard real–time predictability (unlike the said case of the overdimensioned N-GREEN in the metro aggregation), a maximisation of the effective capacity of the network might be understandably desired by the client. In the case of N-GREEN, where nodes are typically supposed to have a single owner, i.e. there are no potentially 'egoistic' nodes, one can think of a number of cooperative packet management schemes, where each node has a substantial liberty in shaping its traffic and, by sending out its statistics, the traffic of the other nodes. N-GREEN requires here a dedicated approach, as on the one hand it has several properties of optical networks not found in other networks (e.g. Ethernet over copper), and on the other hand it does not have the flexibility of other coloured packet optical networks (e.g. an N-GREEN slot can only have either none or all of its wavelengths dropped).

Some proposals of enhancements of N-GREEN exist, that use input buffers and would made the architecture apt for a bursty traffic. These are e.g. the sharing methods of optical slots Ssh-time or Ssh-wavelength [4], with simple queue management schemes. They suffer, though, from inaccuracies of the positioning of optical packets in the fibre and because of that, Ssh-time requires a considerable reduction in the transmission speed (at about 40%). In turn, an application of Ssh-wavelength is not currently possible with the streamlined hardware, and it is unknown if this will change in the future.

Here, we model a single N-GREEN ring in the broadcast-and-select mode (B&S) [4], where a slot can be filled only by a single node and is dropped after its last destination. We deduce and optimise functions which, depending on momentary conditions in the ring, give a good timer expiration value ($t_{\max}$), i.e. a moment at which a node should attempt to unconditionally insert its packets, even if the resulting slot fill ratios were inferior. The functions use either local (i.e. within the same node) or global information about the occupancy of the input buffers.

This paper is organised as follows: Sect. 2 discusses the basic traits of the N-GREEN architecture. As we discuss an adaptive value of $t_{\max}$, we need diverse traffic models to test it – they are presented in Sect. 3. The proposed packet management scheme is described in Sect. 4. It relies on the timer expiration value $t_{\max}$, whose adaptation to the network state is discussed in Sect. 5. The last section concludes the topic.

2 Architecture

N-GREEN uses 10 fixed wavelengths. A single network consists of either one unidirectional ring or two parallel rings, one for each direction. A ring is divided into fixed–size slots of $S = 12500$ bytes each, 1250 bytes per a wavelength. A slot passes a node in 1 µs, which gives a transfer rate of 100 Gbps. A specificity of N-GREEN is that there is only a single semiconductor optical gate/amplifier (SOA) [5] for all wavelengths (see Fig. 1), which stays in contrast to a separate SOA

for every wavelength in the much more expensive POADM technology [6]. This means that a slot can only be cleared in its totality, when all the wavelengths are either transferred or suppressed. The speed of SOA is limited – e.g. it would not be possible to handle separately slots of the size of 3 K bytes each.

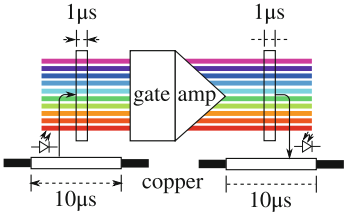


Fig. 1. A symbolic presentation of a N-GREEN node connected to a 10 Gbps Ethernet. There is set of fixed lasers, an optical gate, an amplifier and a receiver.

There is no arbitrariness in the choice of a carrier wavelength – in a single node, a typically 10 Gbps Ethernet card emits or receives traffic, which is converted to/from an optical form by a fixed shift register. Each 10 µs of packets in the electric form can be fitted into 1 µs of a optical signal of all 10 wavelengths, as illustrated in Fig. 1. See that if there is no more than 10 nodes, they can produce up to 100 Gbps of data in a theoretical case (i.e. the bandwidth of the ring). As no packet requires a whole ring, then in the case of a moderately even traffic N-GREEN is potentially overdimensioned (even if there is only a single ring), allowing a laminar data transfer without the need of complex packet buffering and management. This is a good solution for G5 or metro aggregation, where a predictable traffic allows a spatially and temporally balanced load of nodes (e.g. each node connects to a G5 antenna of a similar capacity).

We aim to keep this paper focused (e.g. not complicating it by routing schemes) and due to symmetry considerations, the packet transmission is simulated using only a single ring. However, in Sect. 5.5 we simulate a bidirectional control channel, which reduces the delays due to propagation times, which in turn may potentially increase the effectiveness there.

An N-GREEN network has a single switch node, which connects to other networks. In our model, we just model 10 nodes of the same class and any possible asymmetry is modelled by the traffic patterns.

3 Traffic Models

As we want the network to adapt to different situations, we will use different traffic patterns: an exponential traffic, and two trace-based traffics which use a CAIDA dataset [7], respectively called raw and balanced for the rest of this paper. Within the exponential traffic, in order to accent its smoothness, all packets are of an equal size, being also the mean packet size in the said trace.

In the case of the raw traffic, the anonymised address space [8] is divided into equal parts, one for each N-GREEN node N_i, $i = 1, 2 \ldots N$ for the number of nodes $N = 10$. In order to balance the traffic more, so that its bursty characteristic is still there, but with a more distributed load, we then iteratively relax the said equidistant parts, so that the following penalty equation is minimised:

$$P = \sum_{i=1}^{N} \min\left(D, |\log \frac{v_i}{\bar{v}}|\right), \ \bar{v} = N^{-1} \sum_{i=1}^{N} v_i \tag{1}$$

where v_i is a traffic in bytes emitted or received by N_i, depending on whether the address space to divide is the source or the target one. The logarithm and $D = 10$ together moderate penalty contributions where a node receives a very small or a very large amount of data relatively to the mean, so that their respective importance is not overly dominating over the rest of the contributions to the penalty. This can be important as the division is granular – there are never boundaries between addresses with the same two least important IP bytes, as the probability that such addresses belong to a single local network is elevated, and we want N-GREEN to primarily transport data having relatively distant destinations.

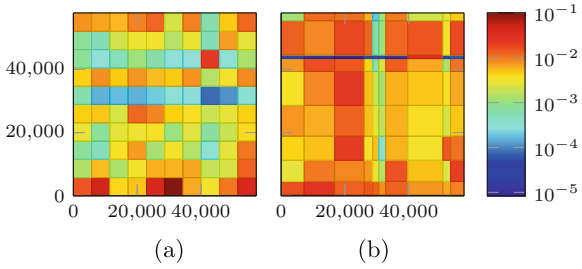


Fig. 2. Normalised histograms of the traffic models (a) raw and (b) balanced, subsequent buckets in each axis are separated by lines and represent subsequent nodes N_i, as emitters for axis x and destinations for axis y. Addresses are represented as coordinates being their 16 bit most–important bits: analogously to the buckets, x- and y- axes show respectively a source and a destination address. See that due to independent balancing of sources and destinations, it is far from ideal. Yet some hot spots has been attenuated, e.g. compare the bottom rows of (a) and (b), where the balancing decreased a large number of packets read by N_1.

Respective normalised histograms of the amount of traffic in bytes for the two trace-based models are shown in Fig. 2. The histograms, begin based on a real trace, show packets with the same origin and destination nodes due to the quantisation of trace addresses to node numbers. It does not make sense that a node sends optical packets to itself, thus these packets are suppressed within the simulation.

Any of these traffic patterns can be further regulated by changing the timescale of arrival times. It is represented by the load

$$L = \frac{\bar{s}}{10\,\text{Gbps}} \tag{2}$$

where s is the average stream rate emitted by a statistical node. In the original metro aggregation architecture with Ethernet cards of 10 Gbps, $L = 1$ would represent a theoretical limit where all the cards transmit data to the ring at their full duty. As in our case we assume a bursty traffic, we allow the input buffers to be filled at a momentary rate larger than 10 Gbps, provided only that the value of L is met. It is an important difference – see that even for $N = 10$ the network

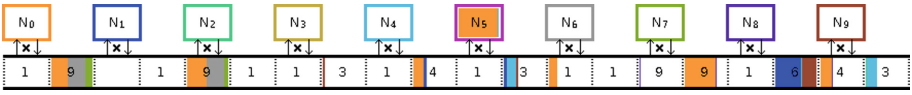


Fig. 3. An example state of input buffers and of the ring (displayed as a bar for brevity). Colours show the number of bytes of packets destined to a given node, a number in the slot depicts the emitter node. Network interfaces symbolise that a node can read a slot, then delete it, then insert new contents into the just–emptied slot. In the visualised state, N_9 produced a burst of slots containing packets to N_6, N_7 and N_8, causing a packet loss in the input buffer of N_5.

can be no longer overdimensioned, as the speed limit of 10 Gbps/100 Gbps = 10% of ring capacity inserted by a single node can be temporarily exceeded, which may overload the ring. See Fig. 3 for an example of a momentary state at $L = 0.4$, where a rare coincidence has led to a temporary congestion.

4 Packet Management

An idea of a timer, which once expired, leads to an unconditional formation of a slot, has been already studied in the case of optical backhaul networks, e.g. in an LTE network [9] where the timer is constant, globally or CoS–wise. In the case of N-GREEN, only a constant expiration time has been studied [4]. Further we will discuss a variable expiration time $t_{\max}$, used in the following buffer management scheme.

Each node has $N - 1$ separate FIFO input queues with electronic packets (EP), one queue for each possible destination. The EPs can be arbitrarily segmented and multiplexed into a flat payload of an OP. To each segment a preamble is added with the organisational data. Given that EPs already contain headers, we concluded that it is enough to have a preamble of 8 bytes, holding values like an unique EP identifier (4 bytes), an offset of the segment to to the beginning of the segmented EP (2 bytes, the 1st segment of a given EP would always have an offset of 0) and finally the length of the segment (2 bytes). A number of

segments, together with their preambles, form an optical packet (OP) which is never larger than a single slot of the mentioned size of $S = 12500$ bytes. The network operates in the B&S mode, and thus a single OP can contain fragments of EPs with different destinations. If an EP is segmented into different OPs, its fragments must be transmitted in the internal order, but they can be interspersed by other payload. Each node looks into every passing OP for segments with fragments of EP with its destination address, recovers these fragments if any and merges them back into EPs. If all the content of a slot has been received at a node N_i, the exact node empties the slot using its optical gate.

If a payload ready to send by a node can form an OP of the size of S, an OP formed from the oldest EPs is sent immediately to the first empty slot available. Otherwise, an oldest EP is found, for which the timer $t_{\max}(N_s, N_t)$ expired, where N_s and N_t are respectively the source and the target node. Fragments of such EPs are iteratively merged until an OP of the size of S can be formed. If that is not possible (i.e. no more EPs with timers expired, but at least a single EP with its timer expired has been found so far, i.e. must be sent), then in order to maximise the usage ratio of the slot, the rest of the OP is formed from any of the oldest EPs left in the buffers. Always a biggest possible fragment of an EP is taken – the only limitation here is the maximum size of an OP.

A new EP which does not fit into its input buffer is immediately considered lost, with its "resend time" of $t_r = 1\,\mathrm{s}$, a large value in comparison to an average insertion time found in N-GREEN, which is normally of the order of tenths of microseconds. Because of the large t_r, lost packets decrease considerably the transmission quality. On the other hand, the assumed t_r seems to be unrealistically small given the reality found in computer networks [10]. This is deliberately so – the said quality will be further maximised using a direct optimisation method, and too large values of t_r might produce gradients in the optimisation space which would be difficult to handle. The low t_r does not seem to be a problem though, if we assume that packet loss ratio should be very low or even that there should be virtually no packet loss in a properly configured N-GREEN ring. Any EP already in the input buffer which is older than $t_r/2$ is also lost, in order to prevent a potential domino effect of packets no longer considered valid by the destination application – a latency of $t_r/2 = 0.5$ s is considered very large in the realm of N-GREEN.

We see that $t_{\max}(N_s, N_t)$ can be potentially sensitive to the state a ring fragment, through which it is transferred. It can be important if e.g. a part of the ring is busy and thus an increased slot fill ratio is recommended in order to maximise resource usage.

5 Adaptation of $t_{\max}$

We will use a direct optimisation method which repeatedly runs simulations in order to evaluate a quality function at different points of the optimisation space. The coordinates of the space are constant coefficients in the equations representing $t_{\max}(N_s, N_t)$.

We can choose if the expiration time of a EP in the input buffer of a node N_i uses only the statistics concerning N_i, or on the contrary, if each node has access to some global statistics about the other nodes. The latter solution is more powerful, at the cost of a need of a transmission of such statistics between the nodes.

5.1 Optimisation Method

We will use a Nelder-Mead optimisation method [11] with a relative convergence criterion of $1 \cdot 10^4$ and a starting point at the origin. The method is not predestined to work with a noise optimisation space, yet possibly due to large A, it turned out to work well, when compared with e.g. [12], theoretically adapted to noise but requiring a bounded space.

In order to avoid overfitting [13] or convergence difficulties, and for practical reasons like a temporal performance of the packet management in a real device, we will avoid "general" equations for $t_{\max}(N_s, N_t)$ where the arguments would be a large set of randomly chosen parameters, whose importance would be largely left to the optimisation process. Rather, the equations will have a concrete justification in relation to the architecture of the network.

5.2 Estimation of a Transmission Quality

The direct optimisation method needs to probe a transmission quality in each iteration. For a single estimation of the transmission quality, we will use one or more simulations, a single simulation is run at single load L. Within each such simulation, statistics of the first $A = 10 \cdot 10^6$ packets are collected (the number of packets generated can be greater, so to provide these A packets normal traffic conditions). The constant A turns out to be high enough for the noise in the gathered statistics to be averaged out sufficiently, given a resistance to noise of the used optimisation method. We have chosen a maximum total size of input queues per node of $B = 10 \cdot 10^6$ bytes, which seems reasonable given the current technology.

As an estimation of the transmission quality, we use the mean latency of an insertion of a packet into the ring. We do not include the travel time through the ring, as this path is all optical and thus predictable and very fast. We assume that the ring contains $M = 1000$ slots at a given time, which is realistic in the metro–area applications of N-GREEN.

Let there be a set of loads $\{L_s : 0.01, 0.1, 0.2, \ldots 0.9, 0.95, 1.0\}$ which cover an usable range of traffic intensities. We use the extra value 0.95 to better sample the region with a high probability of losses, where small differences may in effect lead to very different latencies due to a relatively large t_r. The optimisation process will have two possible modes:

- O_{load}, which finds separately a set of optimised $t^{L}_{\max}(N_s, N_t)$ for each element L of L_s;
- O_{univ}, where a single quality optimisation covers whole L_s and the resulting transmission quality is given by an arithmetic mean of mean latencies for each load in L_s; thus, there is only a single optimised equation $t^{\text{univ}}_{\max}(N_s, N_t)$.

Obviously, the function found by O_{univ} is what is expected in practice: an equation adapted to all loads. O_{load} is not practical, as the mean load is only known during the simulation; it can only be estimated by $t^{\text{univ}}_{\max}(N_s, N_t)$ in a real network. We use O_{load} to estimate, if $t^{\text{univ}}_{\max}(N_s, N_t)$ adapts well to different network conditions. Namely, if a latency for a given $t^{L_s}_{\max}(N_s, N_t)$ is substantially better than that given by $t^{\text{univ}}_{\max}(N_s, N_t)$ for the same load L_s, it means that an estimation of the expiration time could be theoretically more effective in the case of O_{univ}.

Our simulator is written from scratch for N-GREEN, achieving on average 1 million packets transmitted per a second of physical time, yet the convergence speed of the optimisation method still turned out to be a major limiting factor.

5.3 Accumulation of $t_{\max}$

Only a range of values of $t_{\max}$ has a meaning. For example, negative $t_{\max}$ has normally no interpretation and $t_{\max} \geq 100\,\mu$s degrades the quality overly for the standards of N-GREEN – such elevated values were not even considered previously in the case of that network [4].

Let $t_{\max}(N_s, N_t)$ be further constructed as follows

$$t_{\max}(N_s, N_t) = t_{\lim}\left(t_{\text{opt}}(N_s, N_t)\right) \tag{3}$$

where $t_{\text{opt}}(N_s, N_t)$ is one of the optimised equations considered later, and $t_{\lim}(t)$ limits t by $t_l = 0$ and $t_h = 100\,\mu$s, the resulting value being the effective timer expiration. We can use different techniques here. For example, we could just clamp t in order to obtain $t_{\max}$ without doing anything else, but we have two reasons for not doing it:

- it might cause flat quality regions in the optimisation space, as different values of the optimised coefficients might lead to the same effective (clamped) value of $t_{\max}$;
- instead, we could find a new application for these regions – a temporal accumulation of $t_{\max}$.

Let thus each node N_i have its accumulation value C_i, with the following interpretation, allowing an optional inertia when estimating $t_{\max}$:

- negative C_i means that $t_{\max}$ should also be small in some short period in the future, i.e. something occurred which is a strong predictor, that there won't be a need for larger $t_{\max}$ soon;
- on the contrary, positive C_i indicates a strong predictor, that a large $t_{\max}$ will be required for some time.

Let t_{lim} controls $t_{\max}$ as follows:

$$\begin{gathered} C_i' = \max(-I_C, \min(I_C, C_i + t)) \\ t_{\max} = \max(t_l, \min(t_h, C_i')) \\ C_i = C_i - t_{\max}. \end{gathered} \tag{4}$$

We see that $t_{\max}$ consumes a maximal part of C_i, given the limitations t_l and t_h. Additionally, $I_C = 1ms$ limits the inertia of C_i, as we assume that is unsafe to predict the network state for periods larger than I_C.

5.4 $t_{\max}$ as a Function of Local Conditions

Let us begin with a variant of an estimation of $t_{\max}$ further called *local*, which does not have the need of the transmission of node statistics. It will be compared to an optimised constant expiration time, i.e. a variant further referred to as *const*.

Let the expiration time of EPs in N_i be a function of a total size S_i of input buffers in bytes in the local node. We do not know anything about the shape of the function, thus let us start with a 2nd degree polynomial:

$$t_{\max}(S_i) = a_0 + a_1 \frac{S_i}{d} + a_2 \left(\frac{S_i}{d} \right)^2 \tag{5}$$

where the optimised coefficients a_0, a_1, a_2 are common for all nodes and $d = 1 \cdot 10^5$, being 10% of B, is meant to better the performance of the Nelder-Mead method by increasing the isotropy of parameterisation of the optimisation space.

The resulting $t_{\max}$ and insertion latency are shown respectively in Figs. 4 and 5. In the latter figure, we see practically no improvement in latency values for any traffic type, when comparing the variant local to the variant const. As seen in the top row Fig. 4, both variants give similar $t_{\max}$ for all loads. On the contrary, their $t_{\max}$ varies with the traffic model. These two traits together may indicate a performance problem if a complex traffic is used, which can be thought of as a mixture of different traffic patterns, thus requiring different $t_{\max}$. We can thus expect both the variants const and local to perform poorly in e.g. the case of the traffic model raw. It is confirmed when comparing Fig. 5(a) and (b) – O_{univ} works almost as well as O_{load} for the exponential traffic, yet there is a large performance drop for O_{univ} in the case of the traffic model raw, as large as ten times for $L = 0.01$; the traffic model balanced gives here a more moderate performance drop.

5.5 $t_{\max}$ as a Function of Global Conditions

If we compare $t_{\max}$ (Fig. 4) and a mean $\overline{S}$ of S_i over all nodes during a whole simulation (Fig. 6), we see that both are the lowest for the traffic model exponential and highest for the traffic model raw. That similarity may be a hint for making $t_{\max}$ dependent on different values related to buffer sizes.

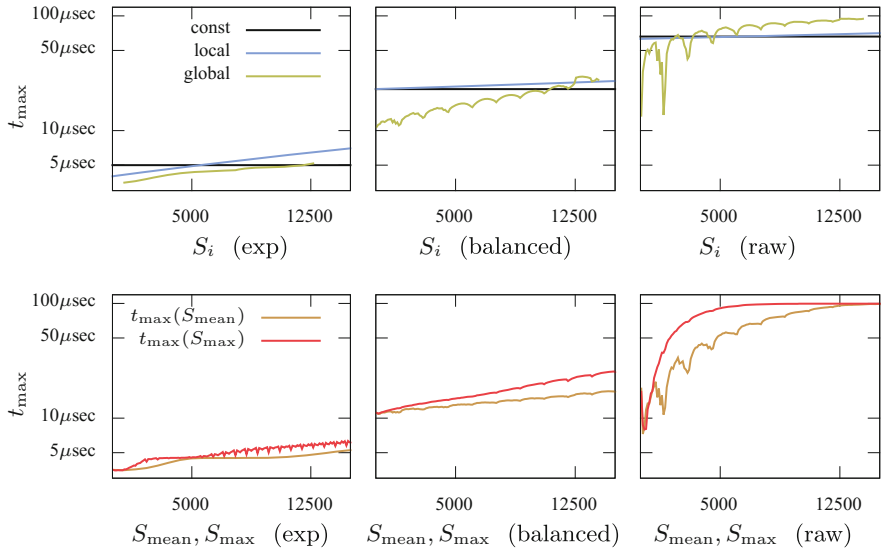


Fig. 4. Timer expiration $t_{\max}$ against sizes of input buffers, optimisation O_{univ}. The top row shows the variants const, local and global against S_i, the bottom row shows the variant global against S_{mean} and $S_{\max}$. For the variant global, only a mean value of an underlying PDF is shown. Columns depict traffic models exp, balanced and raw.

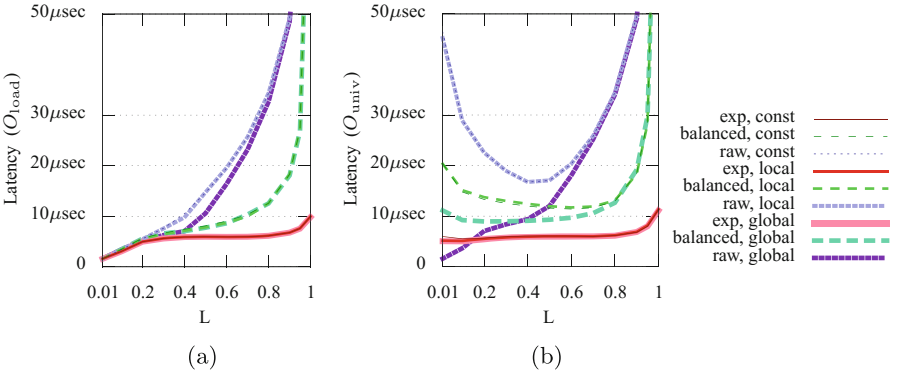


Fig. 5. Latency after optimisations (a) O_{load} and (b) O_{univ}, against L using the variants const, local and global. Latency values out of scale are caused by packet loss.

Let us thus try to use more global sizes of buffers to estimate $t_{\max}$ (a variant *global*). Using queue sizes S_i from different nodes requires their transmission through the network. N-GREEN has a special control channel, where each node can write its momentary total input buffer size. There are obviously delays resulting from the propagation times. We simulate a bidirectional control chan-

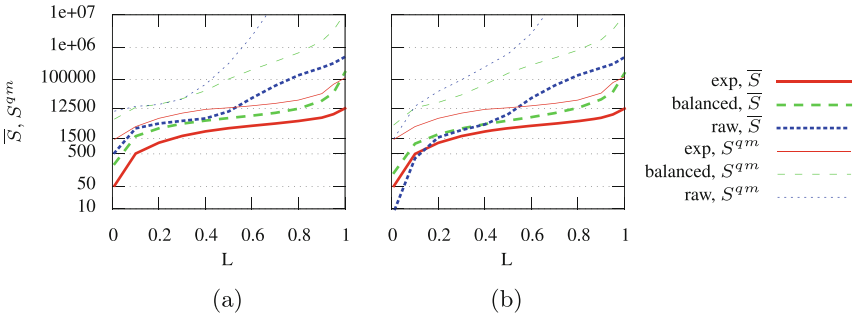


Fig. 6. Dependency between L and S_{mean} for different traffic models, using the variants (a) local and (b) global.

nel, where each node gathers buffers sizes from a ring which provides a smaller propagation time.

To keep the dimensionality of the optimisation space small, we will limit ourselves to a linear equation, adding three values representing the said buffer sizes:

- the already discussed S_i;
- a mean queue size on midway nodes between the source and the target node $S_{\text{mean}}(N_s, N_t)$ where N_s and N_t are as in $t_{\max}(N_s, N_t)$, discussed in the packet management algorithm in Sect. 4;
- a maximum queue size on midway nodes $S_{\max}(N_s, N_t)$.

Let the equation to optimise be as follows:

$$t_{\max}(S_i) = a + a_S \frac{S_i}{d} + a_{\text{mean}} \frac{S_{\text{mean}}(N_s, N_t)}{d} + a_{\max} \frac{S_{\max}(N_s, N_t)}{d}. \quad (6)$$

Looking at latencies Fig. 5, we see the desired effect, as with the exception of the exponential traffic, where the latency was already very low for the variants const and local, the variant global gives a considerable improvement. Additionally, for the traffic model raw, O_{load} and O_{univ} give comparable results.

6 Discussion

The increased unpredictability of a bursty, asymmetric traffic shows in the quality of the results: if the nodes do not share their statistics, it is more difficult to estimate the global situation in the ring within a single node and in effect the quality of the packet transmission decreases. Yet, an unpredictable traffic is exactly the case where a proper packet management is especially important. These factors are seen in the diagrams: as the variability of the traffic increases (i.e. the traffic models exp, balanced and then raw),

– the variant local becomes more and more insensitive to the local buffer size, thus approaching the variant const (Fig. 4, upper row); it shows that S_i alone is an unusable predictor for $t_{\max}$;
– the latency increases and becomes problematic (Fig. 5); this may lead to losses.

The increased performance of the variant global is a result of it being actually sensitive to the available statistics. A four-dimensional surface of (6) may be difficult to visualise, so within the simulation, a mean $t_{\max}$ against S_i, S_{mean} and $S_{\max}$ has been computed, its nonlinearity representing the interdependencies between the three arguments (Fig. 4). We see that for the traffic model raw (the rightmost column in that figure), where the variant global is especially advantageous, the said relations are strong.

Let us estimate a practical advantage of the variants global versus constant. Figure 5 shows an indicator S^{qm}, being the minimum queue size of the topmost 1% of the PDF of S_i (the 999th 1000-quantile) during a whole simulation, i.e. it conveys if there are sporadic congestions that may not show in $\overline{S}$. We see that if the network should be prepared for traffics like the model raw, the load should not exceed 0.6. This is also the value where the expiration time becomes less important, as the criterion of filling a slot is met first with a higher probability due to the more filled input buffers – the three variants in question behave similarly (see Fig. 6(a) vs (b) for larger loads). Clear differences arise for a bursty traffic at small and moderate loads, where the variant global is capable of sensing how much $t_{\max}$ should be decreased to increase the latency – the value is a compromise between an earlier packet expiration and an increased risk of congestion due to a lower fill ratio of slots. This is the situation where the variants const and local show their inflexibility, which leads to even a tenfold increase of latency, compared to the variant global.

It could be feared that the small expiration times in case of the latter variant are risky – a sudden increase of the traffic intensity, unforeseeable by the adaptation method of $t_{\max}$, might cause a packet loss, whereas the variants const and local, with their larger expiration times on average, would be on the safe side. It might happen for some specific traffic patterns, but when we analyse the mean packet loss for O_{univ}, i.e. the optimisation which produces an adaptation expected to respond well to changes, and also for the most uneven traffic case raw, it turns out that the global variant gives a smaller packet loss ratio for the loads L_s, being $3.21 \cdot 10^{-5}$ vs $4.26 \cdot 10^{-5}$ and $4.24 \cdot 10^{-5}$ for respectively the variants const and local.

As there are often considered two opposite rings in N-GREEN, especially if it is expected that there will be a considerable local traffic of e.g. the type machine–to–machine, in the future we would like to add an adaptive routing, that would decide which of these two rings should be chosen by each packet.

References

1. Ware, C., Chiaroni, D.: Towards WDM slot switching for aggregation access and metropolitan applications: the ANR N-GREEN project. In: 2017 19th International Conference on Transparent Optical Networks (ICTON), pp. 1–4. IEEE (2017)
2. Chiaroni, D., Uscumlic, B.: Potential of WDM packets. In: 2017 International Conference on Optical Network Design and Modeling (ONDM), pp. 1–6. IEEE (2017)
3. Lepers, C., Amar, D., Gillet, F., Chiaroni, D.: On the interest of WDM-colored optical packets in metro aggregation networks. In: Asia Communications and Photonics Conference, Optical Society of America (2017). M3C–5
4. Amar, D., Lepers, C., Gillet, F., Lourdiane, M., Ware, C., Chiaroni, D.: WDM slot sharing of colored optical packets for latency improvement and class of service differentiation. In: 2017 19th International Conference on Transparent Optical Networks (ICTON), pp. 1–4. IEEE (2017)
5. Renaud, M., Keller, D., Sahri, N., Silvestre, S., Prieto, D., Dorgeuille, F., Pommereau, F., Emery, J., Grard, E., Mayer, H.: SOA-based optical network components. In: Proceedings, 51st, IEEE Electronic Components and Technology Conference, pp. 433–438 (2001)
6. Popescu, I.: Evaluation of transparent optical multiplexing techniques in transport networks. Télécom Bretagne; Université de Bretagne Occidentale, Theses, September 2015
7. CAIDA UCSD: Anonymized internet traces 2012 (2012). http://www.caida.org/data/passive/passive_2012_dataset.xml
8. Fan, J., Xu, J., Ammar, M.H., Moon, S.B.: Prefix-preserving IP address anonymization: measurement-based security evaluation and a new cryptography-based scheme. Comput. Netw. **46**(2), 253–272 (2004)
9. Benzaoui, N., Pointurier, Y., Bonald, T., Wei, Q., Lott, M.: Optical slot switching latency in mobile backhaul networks. J. Lightwave Technol. **33**(8), 1491–1499 (2015)
10. Kumar, A.: Comparative performance analysis of versions of TCP in a local network with a lossy link. IEEE/ACM Trans. Netw. (ToN) **6**(4), 485–498 (1998)
11. Nelder, J.A., Mead, R.: A simplex method for function minimization. Comput. J. **7**(4), 308–313 (1965)
12. Hansen, N.: The CMA evolution strategy: a comparing review. In: Towards a New Evolutionary Computation, pp. 75–102 (2006)
13. Cawley, G.C., Talbot, N.L.: On over-fitting in model selection and subsequent selection bias in performance evaluation. J. Mach. Learn. Res. **11**(Jul), 2079–2107 (2010)

The Model of the DWDM Access Node

Slawomir Hanczewski(✉), Maciej Stasiak, and Joanna Weissenberg

Faculty of Electronics and Telecommunications, Poznan University of Technology,
Polanka 3, 60-965 Poznan, Poland
{slawomir.hanczewski,maciej.stasiak,joanna.weissenberg}@put.poznan.pl
http://nss.et.put.poznan.pl

Abstract. The following work presents an analytical model of a queuing system with a non-full-availability server and multi-service traffic. This model is then used for an analysis of a DWDM (Dense Wavelength Division Multiplexing) networks access node. Such an approach to DWDM node modeling enables the basic characteristics of the system to be determined and, in particular, allows the average waiting time for service to be estimated. This is of particular importance to operators that take advantage of DWDM networks to offer real-time services.

Keywords: EIG · Queuing system with a non-full-availability

1 Introduction

Declining prices of services and their growing availability cause a rapid growth of data sent via various types of networks each year [1,2]. Such a situation would not be possible had it not been for the development of access networks, mobile and optical (e.g. FTTH - Fiber to the home) networks in particular. This vast amount of data has to be sent through backbone networks. In order to enable backbone networks to serve their purpose, extensive research is being done on how to increase their capabilities, especially within the context of the more and more popular real time services. This work is being extended in two directions: first of all, to increase the available transmission speed through a construction of super-dense DWDM networks [3] and, secondly, to take advantage of traffic management mechanisms in such a way as to ensure the optimum usage of available resources and the assumed level of service. The latter mechanisms include: resource reservation (e.g. [4,5]), compression (e.g. [6–9]), traffic overflow (e.g. [10–12]), prioritisation (e.g. [13,14]) and queuing (e.g. [15–17]). The application of the aforementioned mechanisms will only be effective following their prior parameterization (i.e. a determination of the conditions in which they will have the optimum influence on the network functioning.) Appropriate analytical models can be thus the basis for the parameterization process.

A number of analytical models for telecommunications systems are to be found in the literature on the subject. From these models, those that deal with non-full-availability systems seem to be of particular interest. The more so that

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 245–257, 2018.
https://doi.org/10.1007/978-3-319-92459-5_20

the research studies that have been conducted over the last years indicate that the contemporary telecommunications systems have the features of non-full-availability systems [11,18,19]. Non-full-availability systems can be characterized as such systems in which calls that arrive at the input to the system can only occupy resources from a given set, smaller than the total capacity of the system. Limitations concerning access to system resources may result from either a particular construction of a telecommunications system (e.g. switching networks, systems with overflows) or from the assumptions adopted for the call arrival function in question (e.g. systems with reservation). The basic analytical models of non-full-availability systems are: limited-availability group that is a model of a group with separated links [20], and Erlang's Ideal Grading [21,22].

As part of further research on non-full-availability systems, the authors of this paper propose in the present article an analytical model of a queuing system with a non-full-availability server. The proposed model is applied for the analysis of a DWDM network node. Since buffers in nodes of this type are non-existent, the assumption is that calls wait for service in their source. Following a determination of the maximum number of waiting calls it is possible, however, to not only determine the basic traffic characteristics for a node (serviced traffic, probability of losses), but also to determine the average waiting time for calls that are to be handled. A determination of the average waiting time is crucial from the point of view of real-time services.

The remaining part of the paper is organized as follows. Section 2 presents the basic information concerning the DWDM network. The proposed model of the queuing system is discussed in Sect. 3. Examples of the results of the application of the model for the analysis of a DWDM network are presented in Sect. 4. Section 5 provides conclusions and briefly sums up the paper.

2 DWDM Systems

Wavelength Division Multiplexing (WDM) systems have been successfully used in backbone networks for years. In order to satisfy the current and future demands for data transmissions, these systems are constantly being developed and improved. And this does not apply only to multiplying the number of wavelengths in a single fiber (multiplexing of fiber optics), but also to a capability of systems to set up optical paths dynamically. With the appearance of new conceptions and the first reconfigurable optical multiplexers of the add-drop ROADM type (Reconfigurable Optical Add-Drop Multiplexer), a milestone in the field has been accomplished [23,24]. Such multiplexers allow the wavelength of the transmitted optical signal to be changed in a case when the device does not have a free wavelength in a demanded direction. Very dense wavelength multiplication and the possibility of wavelength conversion makes it possible that the network's resources, i.e. single wavelength, can be offered to the customer only during a connection event. Due to the data rate, there are no data buffers in the nodes, and as a consequence waiting for a transmission occurs at the source (i.e. on the customer's side). Such an approach greatly simplifies any construction

of DWDM network nodes. With the development of modern access networks (mobile and optical) in mind, such a solution may become a common practice in the future.

An exemplary DWDM network is presented in Fig. 1. The network nodes are ROADM multiplexers that have capabilities for dynamic reconfiguration (dynamic setting up of optical paths). This means that optical paths may have their beginning and end in each of the nodes. Network users can demand a path containing one or more wavelengths to be set up, under the condition that all the wavelengths will be handled by the same fiber. Let us consider then the following situation: it is necessary to set up a path consisting of two wavelengths between nodes A and E (Fig. 1). The capacity of each link in the network under consideration is equal to 6 wavelengths. This path can be set up between the nodes A-B-C-E or A-D-C-E. Figure 1b presents the usage of the available resources in the links. In the presented case, the usage of the individual links makes it impossible to match a connecting path between the chosen nodes. This is so, even despite the fact that nodes A and E still have unused resources. This means that the considered system can be treated as a non-full-availability system. The demanded path will be set up only when the resources in the links between nodes B-C or D-C will be free (some paths between nodes B-C or D-C will be disconnected).

In order to determine the traffic characteristics of such a system it is necessary to determine the loss probability for all offered traffic classes and additionally, in the case of real-time services, the average waiting time. All of these parameters can be determined with the help of an appropriate analytical model. For this particular case, it is a model of a queuing system with a non-full-availability server.

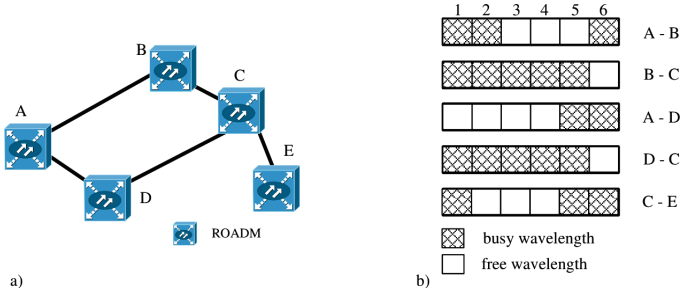


Fig. 1. DWDM network, (a) exemplary network topology, (b) the usage of the available wavelengths in the links

3 Model of a Queueing System with a Non-Full-Availability Server

Analytical models of complex systems with limited access to resources are typically based on Erlang's Ideal Gradings. The structure of this grading and the model of single-service traffic was proposed by Erlang in [21]. According to the concept proposed by Erland, the grading resources[1]. Groups are divided between inseparable sets differing by at least 1 BBU (Basic Bandwidth Unit). A single traffic source has access only to the resources belonging to one set. The number of BBUs belonging to a given set is called the availability d, and the traffic sources which have access to the same BBU set are called the load group g. The number of load groups in an EIG grading is equal to:

$$g = \binom{V}{d}. \tag{1}$$

Limited access to resources is then followed by a situation in which the blocking state within the grading may occur even though there are still free resources in the system. Figure 2 presents a grading with the capacity of $V = 3$ BBUs and availability $d = 2$ BBUs. Figure 2a explains the concept of availability, while Fig. 2b shows the blocking state for load group I. The properties of Erlang's Ideal Grading, both for single-service and multi-service traffic flow are widely discussed in [22].

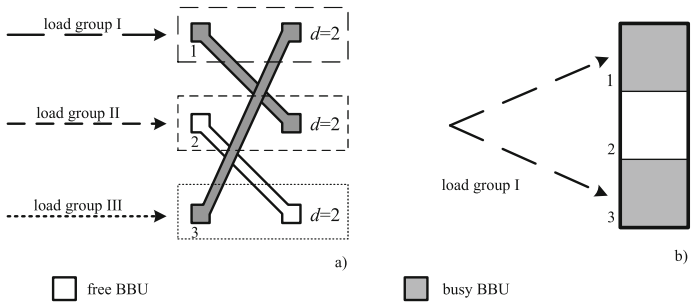


Fig. 2. Erlang's ideal grading (a) the concept of availability, (b) blocking state in load group I

Since limited access to system resources is very frequent in telecommunications, limited access servers should be also considered for queuing systems.

[1] In Erlang's times, the basic system capacity unit was the link that enabled a single voice call to be set up. At present, the system capacity is measured in Basic Bandwidth Units (BBU), and the methods for their determination are described in the literature of the subject. It is generally accepted in a large number of works that a BBU is equal to 1 kbps, and when this is the case the intensity of offered traffic is equal as far as the value and bit rate of a given connection are concerned.

A simplified schematic diagram of such a queuing system is presented in Fig. 3. Here, the assumption is that the server capacity is V BBUs, and that of the queue U BBUs. Let us assume that the system under consideration is offered m traffic classes of the Erlang type that demand $t_1, t_2, ..., t_m$ BBUs to be serviced. The call stream of class i $(0 < i \leq m)$ has a Poisson character with the average intensity equal to λ_i. The service stream of class i $(0 < i \leq m)$ has the exponential character with the average intensity equal to μ_i. The average intensity of offered traffic of class i is equal:

$$A_i = \frac{\lambda_i}{\mu_i} \text{ for } 0 < i \leq m. \tag{2}$$

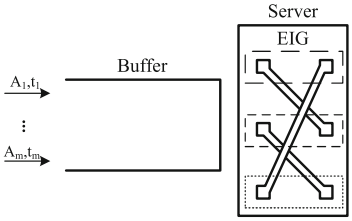


Fig. 3. Queueing system with non-full-availability server

The basis for determining the usage distribution in this non-full-availability queuing system is the Markov service process occurring within such a system. Figure 4 presents a fragment of such a process. The multidimensional Markov process in this system is analysed at the microstate level. Each of the microstates is defined by the number of serviced calls and by the number of calls that have been placed in the queues in each traffic class offered to the system, i.e.

$$(X, Z) = (x_1, \ldots, x_i, \ldots, x_m, z_1, \ldots, z_i, \ldots, z_m), \tag{3}$$

where x_i and z_i denote the number of serviced calls and calls waiting in the buffer of class i, respectively. With the assumption of the reversibility of the service process, we can write for the considered system the following local equilibrium equations between two neighboring microstates of the process:

$$\lambda_i \sigma_i (X - 1_i, Z) [Q (X - 1_i, Z)]_{V,U} = x_i \mu_i \gamma_i' (X) [Q (X, Z)]_{V,U}, \tag{4}$$

$$\lambda_i \beta_i (X, Z - 1_i) [Q (X, Z - 1_i)]_{V,U} = x_i \mu_i \gamma_i'' (X) [Q (X, Z)]_{V,U}, \tag{5}$$

where:

- $[Q(X, Z)]_{V,U}$ denotes the probability of the queuing system under consideration in microstate (X, Z),
- 1_i means one call of class i call,

- $\sigma_i(X, Z)$ defines the probability of admission for service for a call of class i in microstate (X, Z),
- $\beta_i(X, Z)$ denotes the conditional probability of the lack of free resources available in the server for calls of class i in microstate (X, Z),
- the parameters $\gamma_i'(X)$ and $\gamma_i''(X)$ determine the probability of complementary events.

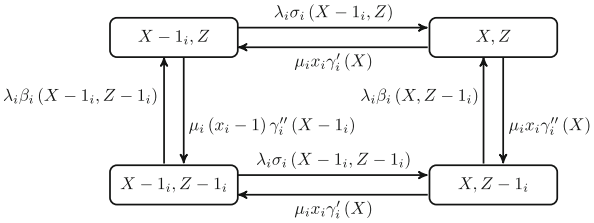


Fig. 4. A fragment of Markov process in queueing system with non-full-availability server

The parameter $\gamma_i'(X)$ determines the conditional probability of an event that, after a termination of service of class i call, none of the class i load group queues have access to the server's free resources. The $\gamma_i''(X)$ parameter determines the probability of an event that at least one queue of class i load groups has access to the released resources, which leads to the admission of the next call of class i for service. Taking into consideration the fact that the probabilities σ_i, β_i, γ_i' and γ_i'' do not depend on the number of serviced calls and those waiting in the queue for each traffic types offered to the grading, but depend on the total number of occupied allocation units, Eqs. (4) and (5) can be rewritten in the following form:

$$\lambda_i\sigma_i(n-t_i, z)\,[Q(X-1_i, Z)]_{V,U} = x_i\mu_i\gamma_i'(n)\,[Q(X, Z)]_{V,U}, \tag{6}$$

$$\lambda_i\beta_i(n, z-t_i)\,[Q(X, Z-1_i)]_{V,U} = x_i\mu_i\gamma_i''(n)\,[Q(X, Z)]_{V,U}, \tag{7}$$

where:

$$\sum_{i=1}^{m} x_i t_i = n \;\wedge\; \sum_{i=1}^{m} z_i t_i = z. \tag{8}$$

Subsequently, after some basic arithmetic transformations and adding up the equations related to a given traffic stream, Eqs. (6) and (7) can be rewritten to the following form:

$$\begin{aligned} x_i t_i [Q(X, Z)]_{V,U} = A_i t_i \Big[& \sigma_i(n-t_i)\,[Q(X-1_i, Z)]_{V,U} + \\ & + \beta_i(n)\,[Q(X, Z-1_i)]_{V,U} \Big], \end{aligned} \tag{9}$$

where $A_i = \dfrac{\lambda_i}{\mu_i}$ denotes the intensity of the traffic of class i. Taking into consideration the independence of the traffic stream offered to the system, we can sum all the equations of the type (9). We will get the following:

$$\sum_{i=1}^{m} x_i t_i [Q(X,Z)]_{V,U} = \sum_{i=1}^{m} A_i t_i \Big[\sigma_i(n-t_i)[Q(X-1_i,Z)]_{V,U} + \beta_i(n)[Q(X,Z-1_i)]_{V,U}\Big]. \tag{10}$$

Taking into consideration (8), Eq. (10) can be written as follows:

$$n[Q(X,Z)]_{V,U} = \sum_{i=1}^{m} A_i t_i \Big[\sigma_i(n-t_i)[Q(X-1_i,Z)]_{V,U} + \beta_i(n)[Q(X,Z-1_i)]_{V,U}\Big]. \tag{11}$$

Summing further, on both sides, Eq. (10) with the set $\Omega(n,z)$, which is a set of all microstates that fulfil condition (8), we get the following:

$$n \sum_{\Omega(n,z)} [Q(X,Z)]_{V,U} = \sum_{i=1}^{m} A_i t_i \sum_{\Omega(n,z)} \Big[\sigma_i(n-t_i)[Q(X-1_i,Z)]_{V,U} + \beta_i(n)[Q(X,Z-1_i)]_{V,U}\Big]. \tag{12}$$

Now, let $[Q(n,z)]_{V,U}$ denote the probability of the macrostate that can be expressed as the sum of all the probabilities of appropriate microstates meeting the condition

$$[Q(n,z)]_{V,U} = \sum_{\Omega(n,z)} [Q(X,Z)]_{V,U}. \tag{13}$$

Taking into consideration the above definition, Eq. (12) can be re-written in the following way:

$$n[Q(n,z)]_{V,U} = \sum_{i=1}^{m} A_i t_i \Big[\sigma_i(n-t_i)[Q(n-t_i,z)]_{V,U} + \beta_i(n)[Q(n,z-t_i)]_{V,U}\Big], \tag{14}$$

where $[Q(n,z)]_{V,U} = 0$, if $n < 0$ and/or $z < 0$, the value $Q(0,0)$ results from the normalisation condition $\sum_{n=0}^{V}\sum_{z=0}^{U} Q(n,z) = 1$. The values of the $\beta_i(n)$ and $\sigma_i(n)$ parameters are determined on the basis of the following relations [22]:

$$\forall_{1\le i\le m} \quad \sigma_i(n) = 1 - \sum_{x=d-t_i+1}^{k} \frac{\binom{d_i}{x}\binom{V-d_i}{n-x}}{\binom{V}{n}}, \tag{15}$$

$$\beta_i(n) = 1 - \sigma_i(n), \tag{16}$$

where:
$k = n - t_i$, if $(d_i - t_i + 1) \leq (n - t_i) < d_i$,
$k = d_i$, if $(n - t_i) \geq d_i$.
Simplifying the assumptions of the model, i.e. assuming that all call classes demand the equal number of BBUs ($\forall_{1 \leq i \leq m} t_i = t = const$), the distribution (14) can be simplified to the distribution [25].

Knowing the occupancy distribution in the queuing system, we can now determine its characteristics. On the basis of the occupancy distribution (14), we can determine the probability of blocking for individual classes of calls on the basis of the following formula:

$$E_i = \sum_{\{(n,z):\ d_i \leq n+z \leq V+U\}} \beta_i(n) \left[Q(n,z)\right]_{V,U}. \tag{17}$$

A determination of the average number of calls of a given class waiting for service requires an analysis of the service process occurring in the considered system at the microstate level. Since a determination of the occupancy distribution at the microstate level is a difficult task, then in order to determine the average number of calls of class i and in (n, z) state, the following approximation is applied: transition from any randomly chosen microstate (n, z) to macrostate $(n, z + t_i)$ is possible when the system does not have a sufficient number of available resources necessary to set up a connection and, at the same time, this call can be admitted to the buffer. When this is the case, the total number of occupied allocation units in the buffer increases by t_i BBU with the probability $\pi_i(n, z)$ that is equal to the ratio between the intensity transferred to the system from state (n, z) to state $(n, z + t_i)$ and the sum of all the streams that cause an increase in the number of calls waiting in the queue by one waiting call:

$$\pi_i(n,z) = \frac{A_i(n)\,\beta_i(n)}{\sum_{i=1}^{m} A_i(n)\,\beta_i(n)}. \tag{18}$$

Therefore, the average number of calls of class i waiting in the buffer can be calculated with the following formula:

$$q_i(n,z) = \sum_{k=0}^{z-1} \left\lfloor \frac{k}{t_i} \right\rfloor \pi_i(n,k). \tag{19}$$

The average number of calls of class i in the queue will therefore be determined by the following formula:

$$q_i = \sum_{z=1}^{U} q_i(n,z)[Q(n,z)]_{V,U}. \tag{20}$$

Knowing the average number of calls of particular classes in the queue, we can determine the average waiting time in the queue:

$$T_i = q_i \frac{1}{\lambda_i}. \tag{21}$$

4 The Modeling of the DWDM Network Node

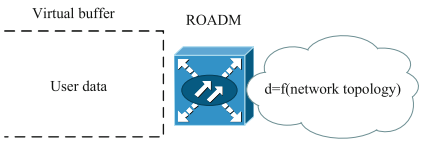


Fig. 5. Virtual buffer in DWDM node

Following the previous assumptions, the proposed model of a queuing system was then used for the analysis of a DWDM network node. For this purpose, the authors assumed the existence of a virtual buffer within the DWDM node with the capacity of U BBUs (Fig. 6), thanks to which it is possible to determine the average waiting time for service (in fact, the data is waiting for transmission on the customer's side). The input data for the model is traffic offered by particular classes of calls to the individual nodes of the considered DWDM network. This data is most often presented in the form of a table, which, for the network presented in Fig. 1, can take on the form presented in Table 1:

Table 1. Class i traffic offered to the DWDM network

Node	A	B	C	D	E
A	x	$A_{i,AB}$	$A_{i,AC}$	$A_{i,AD}$	$A_{i,AE}$
B	$A_{i,BA}$	x	$A_{i,BC}$	$A_{i,BD}$	$A_{i,BE}$
C	$A_{i,CA}$	$A_{i,CB}$	x	$A_{i,CD}$	$A_{i,CE}$
D	$A_{i,DA}$	$A_{i,DB}$	$A_{i,DC}$	x	$A_{i,DE}$
E	$A_{i,EA}$	$A_{i,EB}$	$A_{i,EC}$	$A_{i,ED}$	x

In order to apply the model proposed for the analysis, it is necessary to determine the values of the availability parameters for each serviced call classes. Since this parameter depends on the network structure and load of links between network nodes, the value of availability will depend on the execution of the connection path. The availability parameter values are determined for individual node pairs on the basis of a probability graph [26,27]. The graph for the network from Fig. 1 is presented in Fig. 5. The graph nodes correspond to the network nodes, whereas the edges - to the links between the nodes. For each edge of the graph, the parameter $y_{i,XY}$ was determined (where XY are the respective neighboring network nodes). This parameter determines the probability of the blocking of class i in a given edge. If we assume that we are considering only one class of calls (single-service network), then $y_{i,XY}$ can be interpreted as the link load between nodes X and Y. Knowing the distribution of traffic offered to the individual nodes (for real networks they can be determined on the basis

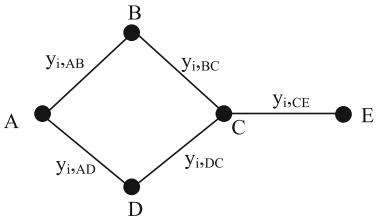


Fig. 6. Probability graph for the considered DWDM network (paths A-E)

of measurements, Table 1), $y_{i,XY}$ values can be determined on the basis of the formula [28,29]:

$$n[P(n)]_{V_{XY}} = \sum_{i=1}^{m} A_{i,XY}[P(n-t_i)]_{V_{XY}}, \tag{22}$$

where, V_{XY} is the capacity of a given link and $A_{i,XY}$ is traffic of class i offered to the link between nodes XY.

After determining the distribution (22), the value of parameters $y_{i,XY}$, can be calculated in the following way:

$$y_{i,XY} = \sum_{n=V_{XY}-t_i+1}^{V_{XY}} [P(n)]_{V_{XY}}. \tag{23}$$

Knowing the value of the $y_{i,XY}$ parameters, we can determine, on the basis of the graph, the probability of the lack of a free paths in the network ($\phi_{i,XY}$):

$$\phi_{i,AB} = y_{i,AB}. \tag{24}$$

For the neighboring nodes, e.g. A-B, the value $\phi_{i,AB}$ is equal to the parameter $y_{i,AB}$.

The $\phi_{i,AE}$ for the A-E path is equal to [30]:

$$\begin{aligned}\phi_{i,AE} = ([1-(1-y_{i,AB})\\ (1-y_{i,BC})][1-(1-y_{i,AD})(1-y_{i,DC})])y_{i,CE}.\end{aligned} \tag{25}$$

Finally, the average number of BBUs (wavelengths) available for a given path (availability) is equal to:

$$d_{i,XY} = (1-\phi_{i,XY})V. \tag{26}$$

4.1 Numerical Examples

Since the model presented in this paper is an approximated one, for its verification the authors used a specially developed DWDM network simulator implemented in C++. The results presented in Figs. 7 and 8 were prepared for a

network with the structure presented in Fig. 1 for data transmission between nodes A and E. It was assumed that the capacity of the links between the nodes was equal to 32 wavelengths. The network was offered $m = 3$ call classes that demanded $t_1 = 1$, $t_2 = 2$ and $t_3 = 4$ wavelengths. Another assumption in the experiment was that traffic offered by particular classes of calls satisfied the condition $a_1t_1 : a_2t_2 : a_3t_3 = 1 : 1 : 1$. The obtained results are presented in Figs. 7 and 8 as the function of traffic offered to one BBU in the server, i.e. a single wavelength. Another assumption was that it was possible to buffer the calls in the source node, setting the total at $U = 6$ wavelengths.[ht] Fig. 7 shows the values of the blocking probability for all offered call classes, while Fig. 8 presents the average waiting times.

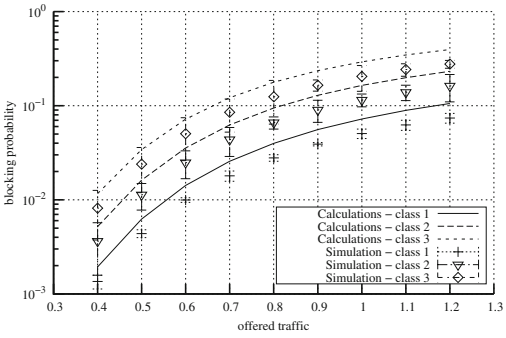


Fig. 7. Blocking probability

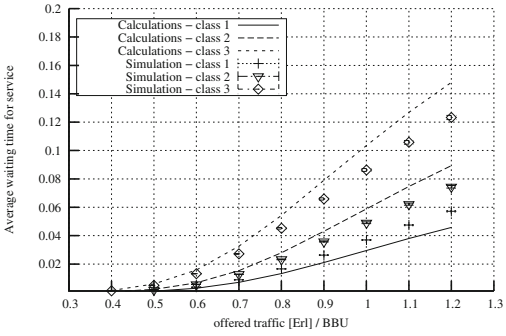


Fig. 8. Average waiting time for service

Each point of the plot shown in Figs. 7 and 8 denotes respectively the average value of blocking probabilities and average waiting time, obtained in 10 series. In particular, simulation series 10^5 of the incoming calls of the "oldest" class were offered. The results of the simulations are shown in the charts in the form of marks with the 95% confidence intervals calculated after the t-Student distribution. The presented results confirm satisfactory accuracy (from the engineer point of view) of proposed model.

5 Conclusion

This paper proposes a new model of a queuing system with non-full-availability server and multi-service traffic. This model is used for an analysis of a DWDM network node. Owing to the introduction of the notion of the virtual buffer, it has become possible to not only determine the probability of losses in the node, but also the average waiting time for service. Therefore, the paper fits into the trend in the research into the application non-full-availability analytical models for modeling of telecommunications systems. The presented results confirm the validity of the assumptions adopted in the study.

Acknowledgements. This article was develop as a result of the research project 2016/23/B/ST7/03925 entitled "Modeling and service quality evaluation of Internet-based services" funded by the National Science Centre.

References

1. Ericsson mobility report on the pulse of the networked society. Technical report, Ericsson (2016)
2. Cisco visual networking index: Forecast and methodology, 2015–2020 white paper. Technical report, CISCO (2016)
3. ITU-T: Spectral grids for WDM applications: DWDM frequency grid. Technical report, Recommendation ITU-T G.694.1 (2012)
4. Stamatelos, G.M., Koukoulidis, V.N.: Reservation-based bandwidth allocation in a radio ATM network. IEEE/ACM Trans. Netw. **5**(3), 420–428 (1997)
5. Stasiak, M., Głąbowski, M.: Point-to-point blocking probability in switching networks with reservation. In: Proceedings of 16th International Teletraffic Congress, vol. 3A, Edinburgh, UK, Elsevier, pp. 519–528, June 1999
6. Räcz, S., Gerö, B.P., Fodor, G.: Flow level performance analysis of a multi-service system supporting elastic and adaptive services. Perform. Eval. **49**(1–4), 451–469 (2002)
7. Moscholios, I., Logothetis, M., Kokkinakis, G.: Connection-dependent threshold model: a generalization of the Erlang multiple rate loss model. J. Perform. Eval. **48**, 177–200 (2002)
8. Sobieraj, M., Stasiak, M., Weissenberg, J., Zwierzykowski, P.: Analytical model of the single threshold mechanism with hysteresis for multi-service networks. IEICE Trans. Commun. **E95–B**(1), 120–132 (2012)
9. Moscholios, I.D., Logothetis, M.D., Boucouvalas, A.C.: Blocking probabilities of elastic and adaptive calls in the Erlang multirate loss model under the threshold policy. Telecommun. Syst. **62**(1), 245–262 (2016)
10. Huang, Q., Ko, K.T., Iversen, V.B.: Approximation of loss calculation for hierarchical networks with multiservice overflows. IEEE Trans. Commun. **56**(3), 466–473 (2008)
11. Głąbowski, M., Hanczewski, S., Stasiak, M.: Modelling of cellular networks with traffic overflow. Math. Probl. Eng. **2015**(11), 1–15 (2015). Article ID 286490
12. Głąbowski, M., Kaliszan, A., Stasiak, M.: Modelling overflow systems with distributed secondary resources. Comput. Netw. **108**, 171–183 (2016)

13. Subramaniam, K., Nilsson, A.A.: Tier-based analytical model for adaptive call admission control scheme in a UMTS-WCDMA system. In: 2005 IEEE 61st Vehicular Technology Conference, vol. 4., pp. 2181–2185, May 2005
14. Hanczewski, S., Stasiak, M., Zwierzykowski, P.: Modelling of the access part of a multi-service mobile network with service priorities. EURASIP J. Wirel. Commun. Netw. **2015**(1), 1–14 (2015)
15. Hanczewski, S., Stasiak, M., Weissenberg, J.: A queueing model of a multi-service system with state-dependent distribution of resources for each class of calls. IEICE Trans. Commun. **E97–B**(8), 1592–1605 (2014)
16. Stasiak, M.: Queuing systems for the internet. IEICE Trans. Commun. **E99–B**(6), 1224–1242 (2016)
17. Hanczewski, S., Kaliszan, A., Stasiak, M.: Convolution model of a queueing system with the cFIFO service discipline. Mob. Inf. Syst. **2016**(8), 15 (2016). Article ID 2185714
18. Hanczewski, S., Sobieraj, M., Stasiak, M.D.: The direct method of effective availability for switching networks with multi-service traffic. IEICE Trans. Commun. **E99–B**(6), 1291–1301 (2016)
19. Głąbowski, M., Hanczewski, S., Stasiak, M.: Modelling load balancing mechanisms in self-optimising 4G mobile networks with elastic and adaptive traffic. IEICE Trans. Commun. **E99–B**(8), 1718–1726 (2016)
20. Stasiak, M.: Blocking probability in a limited-availability group carrying mixture of different multichannel traffic streams. Annales des Télécommunications **48**(1–2), 71–76 (1993)
21. Erlang, A.K.: The application of the theory of probabilities in telephone administration. Scandinavian H.C. Orsted Congress (1920)
22. Głąbowski, M., Hanczewski, S., Stasiak, M., Weissenberg, J.: Modeling Erlang's ideal grading with multi-rate BPP traffic. Math. Probl. Eng. **2012**, 35 (2012). Article ID 456910
23. Vreeburg, C.G.M., Uitterdijk, T., Oei, Y.S., Smit, M.K., Groen, F.H., Metaal, E.G., Demeester, P., Frankena, H.J.: First InP-based reconfigurable integrated add-drop multiplexer. IEEE Photonics Technol. Lett. **9**(2), 188–190 (1997)
24. Neilson, D.T.: Photonics for switching and routing. IEEE J. Sel. Top. Quantum Electron. **12**(4), 669–678 (2006)
25. Thierer, M.: Delay system with limited accessibility. In: Prebook of the 5th International Teletraffic Congress, pp. 203–213 (1967)
26. Lee, C.: Analysis of switching networks. Bell Syst. Techn. J. **34**(6), 1287–1315 (1955)
27. Le Gall, P.: Etude du blocage dans les systemes de commutation telephonique automatique utilisant des commutateurs electroniques de type crossbar. Annales des Télécommunications **11**(7–9) (1956). **159**(7/8) and **180**(9)
28. Kaufman, J.: Blocking in a shared resource environment. IEEE Trans. Commun. **29**(10), 1474–1481 (1981)
29. Roberts, J.: A service system with heterogeneous user requirements – application to multi-service telecommunications systems. In: Pujolle, G. (ed.) Proceedings of Performance of Data Communications Systems and their Applications, Amsterdam, North Holland, pp. 423–431 (1981)
30. Stasiak, M.: An approximate model of a switching network carrying mixture of different multichannel traffic streams. IEEE Trans. Commun. **41**(6), 836–840 (1993)

A Queueing Model of the Edge Node in IP over All-Optical Networks

Kuaban Godlove Suila[1], Tadeusz Czachórski[2(✉)], and Artur Rataj[2]

[1] Institute of Informatics, Silesian Technical University,
ul. Akademicka 16, 44–100 Gliwice, Poland
gkuaban@yahoo.com
[2] Institute of Theoretical and Applied Informatics, Polish Academy of Sciences,
ul. Baltycka 5, 44–100 Gliwice, Poland
{tadek,artur}@iitis.pl

Abstract. The high demand for more bandwidth and high speed networks stimulates research in the design, optimization and performance evaluation of IP over all-optical networks. In this paper we study optical packet filling algorithm applied at ingress nodes of an all-optical network. Arriving electronic packets of variable sizes are stored at a buffer the volume of which is equal to the fixed size of optical packet. When the available space in the buffer is less than the size of an arriving packet, the stored already content of the buffer is sent inside the optical packet and the electronic packet is rescheduled for the next filling cycle. To avoid excessive delays, the optical packet is dispatched also after a specified deadline whatever is the (non-null) content of the buffer. The performance metrics we consider comprises the filling of optical packets (that means the ratio of blocks in the packet to the actual constant size of the packet) and the distribution of optical packets interdeparture times. The paper demonstrates that the variability of the size of arriving packets and the self-similarity of the input traffic have a visible impact on the both parameters.

1 Introduction

Telecommunication and computer network traffic is growing rapidly at an exponential rate, this is as a result of the increased demand for voice, video and data services. The amount of traffic transported on telecommunication backbone networks will likely double with the increased deployment of Internet of Things (IoT) technologies where it is predicted that by 2020 about 50 billion IoT devices will be connected to the Internet [1]. The high demand for more bandwidth and high speed networks motivates research in the design, optimization and performance evaluation of IP over all-optical networks.

In optical networks signals are transmitted by optical fibres between electronical nodes. The throughput of these nodes is inferior to the throughput of the fibres, therefore they become the bottlenecks of the network. In all-optical networks which are still a technology under development, switching is performed

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 258–271, 2018.
https://doi.org/10.1007/978-3-319-92459-5_21

by optical nodes and the optical signal is not inverted to electronic form along the whole way through the network. This may improve substantially the network performance. However, it is not easy to achieve. Electronic nodes are able to queue and prioritize packets, to decide the further routing, while the optical ones are much simpler and may only, if needed, delay the signal in special fibre loops.

The all-optical transmissions are easier to be organised in a synchronous network transmitting fixed size optical packets where, as there is no enough time nor technical means to perform more extensive functions, data flow through the network without being interpreted in switches. From the access network point of view, it is seen as a tunnel providing transport services. Mapping electronic class of service (CoS) and destination address to the optical packet should be done in an edge (ingress) router. The structure of this router is presented in Fig. 1. It is composed of electronic and optical parts and receives packets delivered by various types of electronic networks.

Incoming electronical packets are first sorted by destination, then by class of service (e.g. CoS1, CoS2, Best Effort) and finally cut by shaper S into blocks (cells) of fixed size. Blocks are filling buffers, each buffer corresponds to a defined destination and class of service. The size of the buffer is the same as the size of optical packet. All blocks originating from an electronic packet should be placed in one optical packet. When an electronic packet arrives and its size is smaller than the space still available the buffer, it is inserted into the buffer. Otherwise, the content of the buffer is framed into an optical packet and dispatched. The last packet starts the next filling cycle. To avoid excessive delays, the content of the buffer is sent after a specified deadline, even if the optical packet is not entirely filled. This solution was discussed already two decades ago by KEOPS (Keys to Optical Packet Switching) project [2,3] but it is still being deployed. Here, we investigate the behaviour of a single buffer assembling packets having the same destination and the same class of service.

A study of a similar problem, based on a simulation model and self-similar traffic represented by a discrete time Markov chain (DTMC) was previously presented in [4]. The choice of DTMC was motivated by the assumption of synchronous ATM networks at the electronic side. We use continuous time Markov chains (CTMC) to model the influence of the type of traffic (Poisson and self-similar) and the size of the electronic packets on the filling of optical packets and on their interdeparture times. The use of CTMC corresponds better to the description of IP traffic. We use Markovian approach in parallel with simulation. This is redundant and motivated by mutual checking of results. The only difference is that in simulation the deadline to send a packet is fixed and in Markov model this constant time is approximated by 10-th order Erlang distribution.

The rest of the paper is organised as follows: Sect. 2 presents the Markov model, Sect. 3 discusses numerical issues of its use, and summarizes simulation model, Sect. 4 presents a few numerical examples illustrating the performance of the buffer. Section 5 concludes the article.

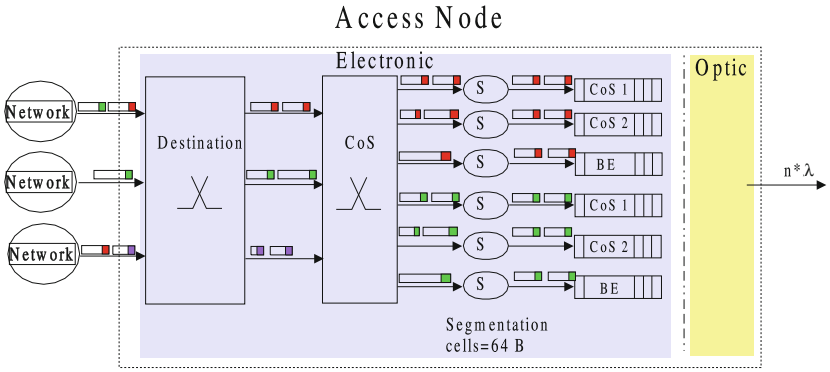


Fig. 1. Electronic-optical access node scheme, source [3]

2 The State Space Model of the Buffer

Suppose that a Poisson stream (i.e. the interarrival times are exponentially distributed) having rate λ of one-block electronic packets arrives to the buffer and the fixed deadline time is represented by P-order Erlang distribution where each of P phases is exponentially distributed with parameter μ. Let N be the size of the buffer expressed in blocks. The considered process of the buffer filling can be modelled as a two-dimensional Markov process $\{N(t), J(t)\}$ on the state space given by:

$$\{N(t), J(t)\} = \{(i, j)|0 \leq i \leq N, 1 \leq j \leq P\} \tag{1}$$

where process $N(t)$ counts the number of blocks arrived to the buffer during $(0, t]$ i.e. present at the buffer at time t while process $J(t)$ counts the phase of the clock.

Figure 2 presents the transition-rate diagram of this process. The diagram determines the Chapman-Kolmogorov equations which solved numerically give state probabilities. The behaviour of the system can be summarised as follows:

1. Initially the buffer is empty and its state is $(0, 1)$.
2. If an electronic packet arrives with the rate λ, the system jumps from state (n, p), $i < N - 1$, to state $(n + 1, p)$.
3. If the system is in a state $(N-1, p)$ and a block arrives, the full optical packet is sent and the Markov chain jumps to the state $(0, 1)$ starting the next filling cycle.
4. If the system is in state $(n, p), p < P$ and the phase of the clock expires with intensity μ, the system jumps to the state $(n, p + 1)$.
5. If the system is in the state (n, P), $n > 0$ (there is no need to send an empty buffer after the deadline) and the last phase of the clock expires with intensity μ, the system jumps to state $(0, 1)$ and an optical packet of n blocks is sent starting the next filling cycle.

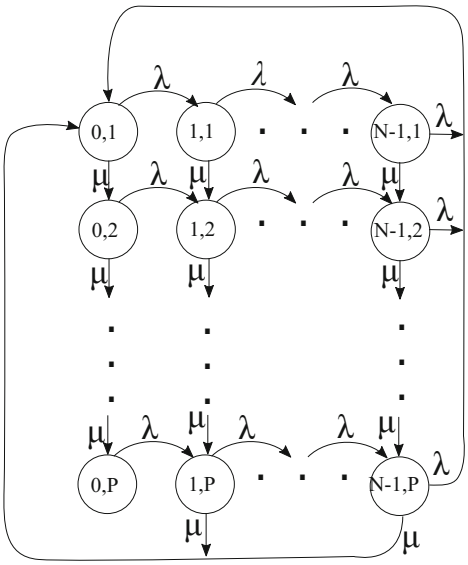


Fig. 2. The transition-rate diagram of the model with one-block arrivals

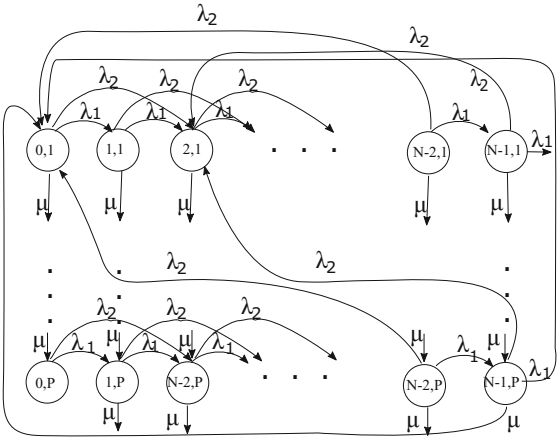


Fig. 3. The transition-rate diagram of the model with one- and two-block arrivals

Generally, at each state two events are possible: a new block arrives with intensity λ or a phase of Erlang distribution is ended with intensity μ. Probability that the state is left due to the first event is $\lambda/(\lambda+\mu)$ and that it is left due to the second is $\mu/(\lambda+\mu)$. If the system is at a state $(N-1,p)$ – the right column of the diagram – the arrival of a new block means that the full packet is sent; if the system is at a state (n,P) – the lowest line of the diagram – the

end of the last phase of the Erlang distribution means that a packet having n blocks is sent. To determine the distribution of the number of blocks present in the dispatched optical packet we should find the conditional probability of n blocks in the packet, provided that the packet is sent. Therefore our normalising constant referring to the sending a packet with any number of blocks is

$$G = \sum_{n=1}^{N-1} \Pr(n, P)\frac{\mu}{\lambda+\mu} + \sum_{p=1}^{P} \Pr(N-1, p)\frac{\lambda}{\lambda+\mu},$$

probability that full packet of N blocks is sent

$$p(N) = \frac{1}{G}\left[\sum_{p=1}^{P-1} \Pr(N-1, p)\frac{\lambda}{\lambda+\mu} + \Pr(N-1, P)\right]$$

and probability of $0 < n < N$ blocks in a sent packet

$$p(n) = \frac{1}{G}\Pr(n, P)\frac{\mu}{\lambda+\mu}.$$

To determine the density of the interdeparture times we consider the distributions of first passage times from the initial state $(0, 1)$ to any state leading to the dispatching a packet, knowing that the distribution of the sojourn time at each state is exponentially distributed with parameter $\lambda + \mu$, and taking these distributions of first passage time with weights corresponding to probability of the paths. Computational aspects of this approach are discussed in [5].

This approach is easily extended to any distribution of the size of arriving electronic packets, e.g. Fig. 3 presents the diagram in the case of one-block packets arriving with the rate λ_1 and two-blocks packets arriving with the rate λ_2. For more complex cases it is hard to present the corresponding diagram but the behaviour of the Markov model is straightforward. If we consider one-, two-, ... n-blocks arrivals with intensities $\lambda_1, \lambda_2, \ldots, \lambda_n$, we should remember that the initial states are $(0, 1), (0, 2), \ldots (0, n)$, the jumps from a state (i, j) are performed with the intensity λ_k to the state $(i + k, j)$ if the the space constraints allow it, i.e. if at a state $(N - i, j)$ the size k of an arriving electronic packet exceeds the free remaining space of i blocks, then the sent optical packet has $N - i$ blocks and the initial state of the next cycle is $(k, 1)$. We should also remember that the number of events that lead to a transition out of a state is larger, e.g. if the chain is in a state (n, P), the probability that the state if left because of the end of the last clock phase (and the optical packet of n blocks is sent) is $\mu/(\mu + \lambda_1 + \ldots + \lambda_n)$.

In the case of non-Poisson arrivals forming self-similar process [6] we used a system of $d = 4$ ON-OFF sources, as it was proposed in [7] with parameters computed in [8] for various Hurst parameters representing the degree of self-similarity. Let us recall that a stochastic process $X(t)$ is self-similar with Hurst parameter H, if for a positive factor g the process $X(gt)/g^H$ has the same stochastic behaviour as the original process $X(t)$, in particular:

$$E[X(t)] = \frac{E[X(gt)]}{g^H}, \qquad Var[X(t)] = \frac{Var[X(gt)]}{g^{2H}},$$

$$R_x(t,s) = \frac{R_x(gt,gs)}{g^{2H}} \quad \text{where} \quad R_x(t,s) = E[X(t)X(s)].$$

Each of applied ON-OFF processes may be parametrized by two square matrices:

$$\mathbf{D_0}^i = \begin{bmatrix} -(c_{1i}+\lambda_{1i}) & c_{1i} \\ c_{2i} & -(c_{2i}+\lambda_{2i}) \end{bmatrix}, \qquad \mathbf{D_1}^i = \begin{bmatrix} \lambda_{1i} & 0 \\ 0 & \lambda_{2i} \end{bmatrix}, \quad i = 1, \ldots d.$$

The element c_{1i} is the transition rate from state ON to OFF of the i-th source, and c_{2i} is the rate out of state OFF to ON, λ_{1i} is the traffic rate when the i-th source is ON. In the state OFF a source is inactive. The sum of $\mathbf{D_0}^i$ and $\mathbf{D_1}^i$ is an irreducible infinitesimal generator $\mathbf{Q}^i$ with the stationary probability vector $\boldsymbol{\pi}_i$

$$\boldsymbol{\pi}_i = \left(\frac{c_{2i}}{c_{1i}+c_{2i}}, \frac{c_{1i}}{c_{1i}+c_{2i}} \right)$$

The superposition of these two-state Markov chains is a new Markov process with 2^d states and its parameter matrices, $\mathbf{D_0}$ and $\mathbf{D_1}$, can be computed using the Kronecker sum of those of the d two-state processes [9]:

$$(\mathbf{D}_0, \mathbf{D}_1) = \left(\oplus_{i=1}^{d} \mathbf{D_0}^i, \oplus_{i=1}^{d} \mathbf{D_1}^i \right)$$

The state space of the whole Markov chain is extended to the description of the state of the four on-off sources, hence the number of states is increased 16 times.

3 Numerical Analysis Using Probabilistic Mmodel Checking

Markovian queueing models have analytic closed form solutions only in special cases. It is related to so-called local balance property, when global balance equations including all events leading to enter or leave a state may be split into local balance equations reflecting arrival and departure of the same customer to a station while all other customers do not move. The most general model based on this principle is BCMP model [10]. Other models use numerical solutions of Chapman-Kolmogorov equations which are global balance equations. This is not easy. The number of states, hence the number of equations to be solved increases rapidly with the complexity of models. One million of equations is nowadays a modest example. The systems of equations are usually ill-conditioned. Therefore the numerical solution of Markov models has long traditions and is well founded discipline, [11,12]; some contributions exist also in Polish literature, e.g. [13,14]. The state transition matrix is sparse, therefore iterative methods which do not change the matrix during iterations (that allows to store only non-zero elements) are favoured. Especially projections methods are recommended. In a general projection technique the original matrix problem of size M is approximated by one of dimension m, typically much smaller than M. A particularly successful class of techniques based on this principle is that of Krylov subspace methods where the

solution is approximated by the vector in a Krylov subspace with minimal residual [15]. The Arnoldi iteration is used to find this vector (generalized minimal residual method, GMRES [16]). Like other iterative methods, GMRES is usually combined with a preconditioning method in order to speed up convergence [17].

Another important problem is to write the equations to be solved. With the increase of the number of states and transitions it is not possible to do it manually. Therefore a number of software tools was proposed, e.g. Markov solver in QNAP [18], XMARCA [11], PEPS [19], PRISM [20], OLYMP [21]. They include not only numerical solvers but also a high level language to formulate the model to be solved, allowing thus automatic generation of the transition matrix. The authors have their own implementation of GMRES but have chosen PRISM because its language is more flexible and allowing to pose questions not only on state probabilities but also on the distribution of time elapsed to the occurrence of a specified event. It is a probabilistic model checker [22,23] developed at universities of Oxford, Birmingham and Glasgow, and probabilistic model checking is a well known technique used in studying systems that exhibit stochastic behaviours and aims to determine the probabilities of the occurrence of certain events during the execution of the system.

To simplify the use of PRISM while computing the densities of interdeparture time distribution, we introduced two fictitious states called the fork state and the absorbing state as shown in Fig. 4 for the case of one- and two-blocks arrivals. The interdeparture time is the first passage time from the fork state to the absorption state: the sum of all sojourn times between the fork and absorbing states, considering all possible paths and their probabilities. We found it convenient to determine the probability of an initial state on the basis of flows entering this state which in turn are given by steady-state balance equations,

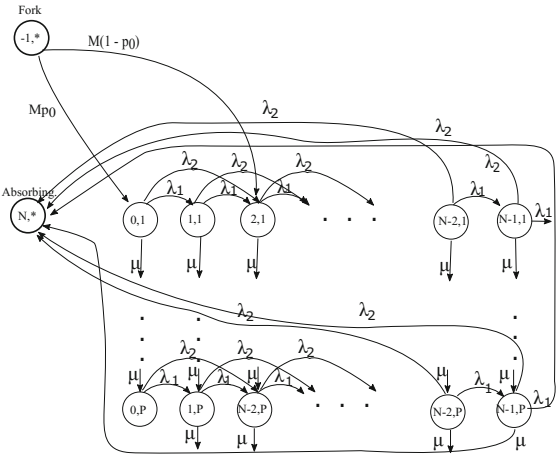


Fig. 4. The transition-rate diagram of the model with fork and absorption states, case of one- and two block arrivals

e.g. if the size of arriving packets is between 1 and n blocks then the initial states are $(0,1)\ldots(n,1)$ and the balance equation for the state $(0,1)$ is

$$\sum_{i=1}^{n}\sum_{j=1}^{P}\Pr(N-i,j)\lambda_i+\sum_{i=1}^{N}\Pr(i,P)\mu=\Pr(0,1)(\sum_{i=1}^{n}\lambda_i+\mu). \tag{2}$$

The left side of this equation gives the flow f_1 entering this state and increasing its probability, the right side gives the same flow f_1 leaving the state and diminishing its probability. The first term (double sum) determines the flow of probability entering this state due to the dispatching full optical packet after arrival of a packet having $i=1,\ldots,n$ blocks to the buffer having exactly this amount of free space. The second term is due to dispatching a packet having $i=1,\ldots N$ blocks because of the deadline. The right hand term gives the intensity of leaving the state $(0,1)$ due to arrivals of packets of all sizes or the end of the first phase of the clock. In the similar way we may consider flows $f_2,\ldots,f_n$ for other initial states and the probability of the choice of an initial state $(i,1)$ is $p_i=f_i/f_1+f_2+\cdots+f_n$. Then PRISM is used to compute the probabilities of the first passage times from the fork state (when the filling starts) to the absorbing state (when packet is dispatch). We specified our model using PRISM language, which is a state-based language, based on the reactive module formalism of Alur and Henzinger [23]. The properties of probabilistic models are expressed by Probabilistic Computation Tree Logic (PCTL) and Continues Stochastic Logic (CSL) which are probabilistic extensions of Computation Tree Logic (CTL). CSL is a temporal logic for describing properties of CTMCs, appropriate to define our model. The principal operators for specifying the properties of stochastic systems include P, S and R operators. The P operator is used to reason about the probability of event occurrence, S concerns the steady-state behaviour of the model and R applies to expected values of model costs and rewards [22,23]. The P operator was used to compute the probabilities of the interdeparture times while the S operator was used to compute the steady-state probabilities of dispatching optical packet of various sizes.

Simulation of the Buffer Using Discrete Event Simulation (DES). Simulation is a software approach to create a virtual environment that emulates the behaviour of a physical system in order to study its performance or properties. We used Discrete Event Simulation (DES) which is the most popular simulation technique used in studying queueing processes. We applied SimPy (Simulation in Python), a library for process-driven discrete event simulation, written in a high level general purpose programming language Python [24]. SimPy utilises Python's object oriented structure to offer a number of objects that represent common structures in a real world system [25]. Processes in SimPy are defined by Python generator functions and may, for example, be used to model the arrival process of packets into the buffer. We implemented buffer and packet objects (properties of the packet are size, time, id. and source). We implemented a process for a Poisson packet generator and another process for an on-off packet

generator (to model self-similar traffic). The simulation experiments validate results of the Markov model.

4 Numerical Examples

Here we present a few results illustrating the impact of the distribution of the size of incoming packets and the type of the input stream on the filling of electronic packets and the distribution of the time between their departures from the node. They have rather qualitative character confirming strong relationship among considered factors.

To roughly imitate the distribution of IP packets we usually assumed two or three sizes of packets expressed in blocks with different granularity: 1,2,3 blocks or 1,10,30 blocks. The arrival intensities of each type of electronic packets are chosen to assure the same average input measured in blocks for each size of packets.

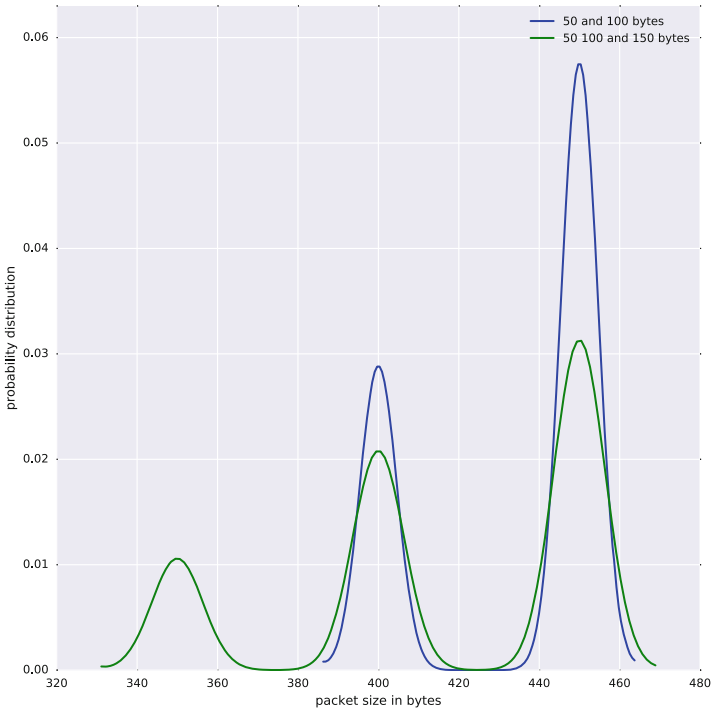


Fig. 5. Distribution of the number of blocks in optical packet. The size of buffer is 10 blocks, the arriving packets are 1, 2 block size or 1, 2, 3 block size. We assume 1 block = 50 bytes, all input flows are Poisson (simulation and numerical results).

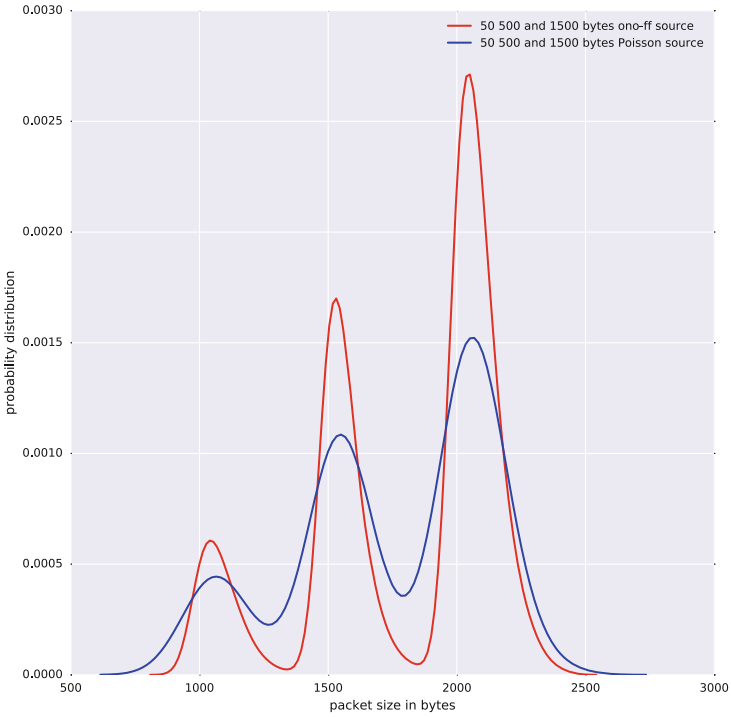


Fig. 6. Distribution of the number of blocks in optical packet for self-similar and Poisson sources (simulation and numerical results)

Figure 5 presents the distribution of the number of blocks inside 10-block optical packets, obtained from discrete event simulation and PRISM model for the buffer size of 10 blocks. The influence of the size of incoming packets on the final filling of the buffer is well visible. Figure 6 shows the influence of self-similarity, Hurst parameter characterising the degree of self-similarity is $H = 0.6$ and the results are compared with Poisson traffic ($H = 0.5$).

The distributions of the interdeparture times (first passage times) obtained from our model are similar to the Erlang distributions obtained by authors in [5] where they used different methods to compute the first passage time distributions in large Markov chains. They are also similar to the interdeparture times obtained in [4] with the use of simulation model. Figure 7 presents the numerical and simulation results that show, in case of Poisson flows, the influence of the sizes of the arriving electronic blocks on the interdeparture times distribution. As electronic blocks of different sizes arrive at the buffer, there are higher chances that the buffer will get full faster or that a packet will arrive and its size is larger than the available space in the buffer, hence the content of

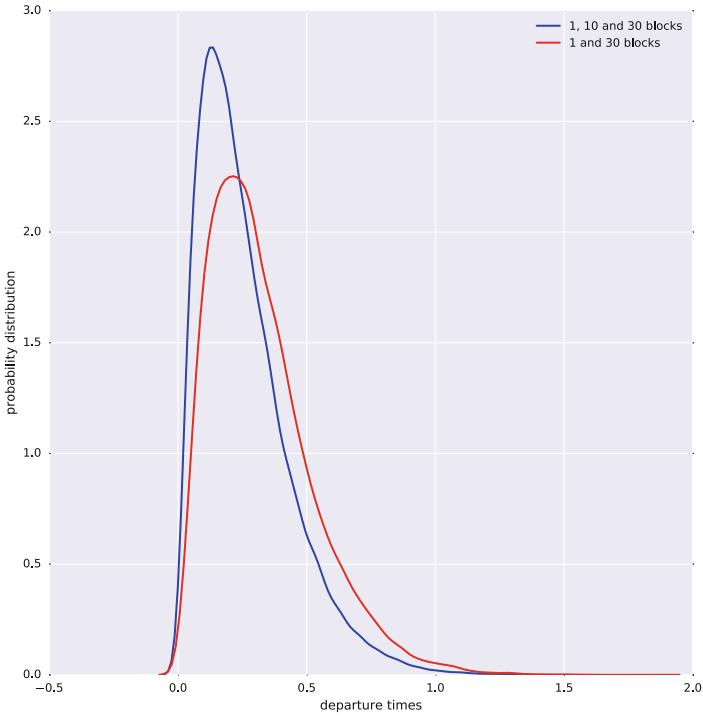


Fig. 7. Distributions of interdeparture times for arriving packets of 1, 10 blocks and 1, 10, 30 blocks (simulation and numerical results)

the buffer is dispatched with shorter interdeparture time. Figure 8 displays the influence of self-similarity of the incoming traffic on the distribution of the interdeparture times. In this particular case the interdeparture times for self-similar traffic have much smaller variance compared to Poisson traffic. This astonishing at first glance result is explained by the fact that in self-similar traffic much frequently optical packets are dispatched because of faster arrival rates during the ON (heavy traffic) periods which are much longer than the OFF (low traffic) periods; it results in faster filling and dispatching (shorter interdeparture times).

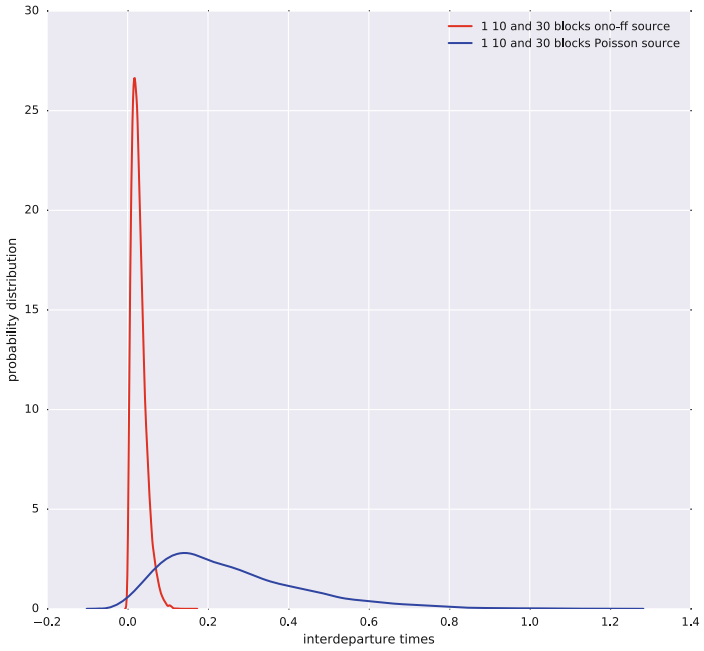


Fig. 8. Distribution of interdeparture times for self-similar and Poisson sources (simulation and numerical results)

5 Conclusions

This introductory study proves that the size of the arriving electronic packets and the pattern of arrivals (Poisson or self-similar traffic) have a visible impact on the traffic inside all-optical network. The distribution of the actual number of blocks in the optical packet has several maxima depending on the size of incoming electronic packets. Even though optical packets are composed of many electronical ones, they leave the router in an irregular way what has negative impact on the performance of all-optical network.

The next step should be a more detailed model taking into account real IP packet distribution and real IP traffic measured in Internet, obtainable e.g. in CAIDA repositories [26]. That will involve much more computational effort.

References

1. Holler, J., Tsiatsis, V., Mulligan, C., Karnouskos, S., Avesand, S., Boyle, D.: From Machine-Machine to the Internet of Things: Introduction to the New Age of Intelligence. Elsevier, Waltham (2014)
2. Gambini, P., et al.: Transparent optical packet switching: network architecture and demonstrators in the KEOPS project. IEEE J. Sel. Areas Commun. **16**(7), 1245–1259 (1998)

3. Kotuliak, I.: Feasibility study of optical packet switching: performance evaluation. Ph.D. thesis, University of Versailles-St-Quentin-en-Yveline (2003)
4. Domanska, J., Kotuliak, I., Atmaca, T., Czachórski, T.: Optical packet filling. In: 10th Polish Teletraffic Symposium PSRT (2003)
5. Harrison, G., Knottenbelt, J.: Passage time distributions in large Markov chains. In: Proceedings of the 2002 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, pp. 77–85 (2002)
6. Park, K., Willinger, W.: Self-Similar Network Traffic and Performance Evaluation. Wiley, New York (2000)
7. Andersen, A.T., Nielsen, B.F.: A Markovian approach for modeling packet traffic with long-range dependence. IEEE J. Sel. Areas Telecommun. **16**(5), 719–732 (1998)
8. Czachórski, T., Domanski, A., Domanska, J., Rataj, A.: A study of IP router queues with the use of Markov models. In: Gaj, P., Kwiecień, A., Stera, P. (eds) Proceedings of Computer Networks 2016, Brunów, Poland, Communications in Computer and Information Science, vol. 608, Proceedings of Computer Networks **2016**, 294–305 (2016). Springer, cham (2016). https://doi.org/10.1007/978-3-319-39207-3_26
9. Fischer, W., Meier-Hellstern, K.: The Markov-modulated Poisson process (MMPP) cookbook. Perform. Eval. **18**(2), 149–171 (1993)
10. Baskett, F., Chandy, M., Muntz, R., Palacios, J.: Open, closed and mixed networks of queues with different classes of customers. J. ACM **22**(2), 249–260 (1975)
11. Stewart, W.J.: An Introduction to the Numerical Solution of Markov Chains. Princeton University Press, Princeton (1994)
12. Stewart, W.J.: Probability, Markov Chains, Queues, and Simulation: The Mathematical Basis of Performance Modeling. Princeton University Press, New Jersey (2009)
13. Bylina, B.: The inverse iteration with the WZ factorization used to the Markovian models. Ann. UMCS Informatica **2**, 15–23 (2004). Lublin
14. Bylina, B., Bylina, J., Karwacki, M.: Computational aspects of GPU-accelerated sparse matrix-vector multiplication for solving Markov models. Theoret. Appl. Inf. **23**(2), 127–145 (2011)
15. Saad, Y.: Krylov subspace methods for solving large unsymmetric linear systems. Math. Comput. **37**, 105–126 (1981)
16. Saad, Y., Schultz, M.H.: GMRES: a generalized minimal residual algorithm for solving non-symmetric linear systems. SIAM J. Sci. Stat. Comput. **7**, 856–869 (1986)
17. Saad, Y.: Precondtioned Krylov subspace methods for the numerical solution of Markov chains. In: Stewart, W.J. (ed.) Computations with Markov Chains, Proceedings of the 2nd International Workshop on the Numerical Solution of Markov chains. Springer, New York (1995). https://doi.org/10.1007/978-1-4615-2241-6_4
18. Potier, D.: New User's Introduction to QNAP2. Rapport Technique no. 40, INRIA, Rocquencourt (1984)
19. PEPS: www-id.imag.fr/Logiciels/peps/userguide.html
20. PRISM - probabilistic model checker. www.prismmodelchecker.org/
21. Pecka, P., Deorowicz, S., Nowak, M.: Efficient representation of transition matrix in the Markov process modeling of computer networks. In: Czachórski, T., et al. Man-Machine Interactions 2, Advances in Intelligent and Soft Computing, vol. 103, pp. 457–464. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23169-8_49

22. Kwiatkowska, M., Norman, G., Parker, D.: Tools Session of Aachen 2001 International Multiconference on Measurement, Modelling and Evaluation of Computer-Communication Systems. Technical report 760/2001, pp. 7–12. University of Dortmund (2001)
23. Hinton, A., Kwiatkowska, M., Norman, G., Parker, D.: PRISM: a tool for automatic verification of probabilistic systems. In: Hermanns, H., Palsberg, J. (eds.) TACAS 2006. LNCS, vol. 3920, pp. 441–444. Springer, Heidelberg (2006). https://doi.org/10.1007/11691372_29
24. TeamSimpy: Simpy Documentation release 3.0.10. MIT (2017)
25. Byrne, N., Geraghty, J., Liston, P., Young, P.: The potential role of open source discrete event simulation software in the manufacturing sector. In: Proceedings of the Operational Research Society Simulation Workshop (2012)
26. https://data.caida.org/datasets/passive-2008/equinix-chicago/20080319/

Multiserver Queueing System with Non-homogeneous Customers and Sectorized Memory Space

Marcin Ziółkowski[1(✉)] and Oleg Tikhonenko[2]

[1] Department of Informatics, Faculty of Applied Informatics and Mathematics, Warsaw University of Life Sciences, Ul. Nowoursynowska 159, 02–787 Warsaw, Poland
marcin_ziolkowski@sggw.pl

[2] Faculty of Mathematics and Natural Sciences, College of Sciences, Cardinal Stefan Wyszyński University in Warsaw, Ul. Wóycickiego 1/3, 01-938 Warsaw, Poland
o.tikhonenko@uksw.edu.pl

Abstract. In the present paper, we investigate a multiserver queueing system with non-homogeneous customers. As non-homogenity, we mean that each customer is characterized by some random l-dimensional volume vector. The arriving customers appear according to a stationary Poisson process. Service time of a customer does not depend on his volume vector and has an exponential distribution. Memory space is composed of l limited parts in accordance with customers volume vectors components. For this system, the steady-state distribution of the number of customers present in the system and loss probability are determined. An analysis of some special cases and some numerical examples are attached as well.

Keywords: Multiserver queueing systems
Queueing systems with non-homogeneous customers
Queueing systems with sectorized memory
Loss probability · Stieltjes convolution

1 Introduction

Models of queueing systems with non-homogeneous customers are used in many areas of computer systems designing. In comparison to the classical queueing systems models [2,11], we additionally assume that arriving customers have some random volume so they usually are characterized by the joint distribution function of service time and customer's volume. In the theory of queueing systems with non-homogeneous customers we investigate both systems with unlimited total volume (which is the sum of the volumes of all customers present in the system) and systems with limited (by value V) total volume. We also take into account the character of dependency between service time and customer's volume. In systems with unlimited memory we obtain the characteristics of the total volume at time instant t or in steady state (if exists).

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 272–285, 2018.
https://doi.org/10.1007/978-3-319-92459-5_22

In more practical cases, when the total volume is limited, the arriving customers can be lost even if there are free servers or free places in the queue. In such situations we obtain the distribution of number of customers present in the system and formulae for steady-state loss probability. These characteristics (that depend on the value V) are very useful when we want to calculate the size of the needed memory. Such approach allows to obtain characteristics of the total volume of all customers present in the queueing system as well as loss characteristics more precisely. Analogous systems were first investigated using the results of the classical queueing models analysis [13,14]. Obtained results differed from the simulation ones, because they did not take into account the character of dependency between the customer's volume and his service time. Many generalizations of the classical queueing models connected with the analysis of the class of non-homogeneous customers may be found in [1]–[12]. The cited papers contain mainly the analysis of the classical $M/M/n/m$, $M/G/n/0$, $M/G/1/\infty$ and processor sharing models generalizations.

Moreover, in many real computer systems, the volume of the arriving customer can be interpreted as a random vector, i.e. it consists of l components that are non-negative random variables. For example, TCP/IP packets contain not only different type data, but also some specific information related to packet's length, its number etc. We can use this additional information to put them in correct order that is necessary to reconstruct the information sent to user. On the other hand, some specific data (mail servers, multimedia packets) are often divided on text, audio, video or attachment parts. This division has an evident sense in the case of many practical applications, especially in Internet packets transmission process. This fact is included in the technical solutions (see e.g. [15,16]). In these patents, packets may be understood as the collections of different data type parts transmitting to dedicated for them two limited buffers. In addition, the volumes of different data type parts are independent.

The purpose of our research is to present a mathematical model of a queueing system in which the l-dimensional total volume of customers present in the system is limited by some constant random vector $\mathbf{V} = (V_1, \ldots, V_l)$, $l = 1, 2, \ldots$. This model is the generalization of analyzed in literature $M/M/n/(m, V)$ queueing model with non-homogeneous customers [1,2]. We additionally take into account that the customer's volume can be composed of different type data parts that are located in the separate limited buffers. The total customers volume in such systems consists of some sectors (parts) preserving data of a different type. Later on, we find the steady-state distribution of number of customers present in the system under consideration and steady-state loss probability. The obtained results are not the same as in the classical $M/M/n/m$ queueing system, because the limitation of every sector of the total volume leads to the additional losses of the arriving customers. The scheme of the analyzed model is presented on Fig. 1.

The paper is organized as follows. Section 2 contains a description of the queueing model and necessary notations. For simplicity, we analyse in detail the case of $n = 2$. In this section, we also define a Markovian process describing the

evolution of the system and the functions characterizing the system behavior. In Sect. 3, we build a system of differential equations for the transient system characteristics and give their steady-state solution. In Sect. 4, we obtain the relation for loss probability. In Sect. 5, we generalize the results obtained in previous sections to capture the case of arbitrary l. In Sect. 6, we consider some special cases of the model together with the numerical results. Section 7 presents conclusions and final remarks.

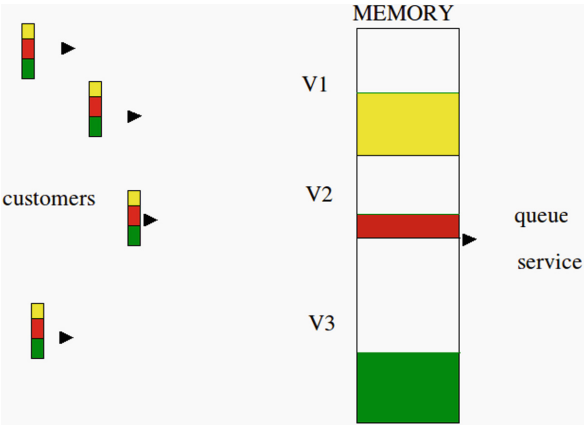


Fig. 1. Model of queueing system with limited and sectorized memory

2 The Model and Notation. Random Process and Functions Describing the System Behavior

In this section, we analyze a modification of the classical $M/M/n/m$ queueing system, in which the arriving customer is additionally characterized by some non-negative two-dimensional random volume vector (ζ_1, ζ_2).

Arriving customers form the Poisson process with constant intensity (entrance flow parameter) a. Let $L(x, y) = \mathsf{P}\{\zeta_1 < x, \zeta_2 < y\}$ be the joint distribution function of the random vector (ζ_1, ζ_2). The volume vectors of different customers are independent. Service time of the customer is independent of his volume vector and has an exponential distribution with parameter μ. The summary volume is also two-dimensional vector $(\sigma_1(t), \sigma_2(t))$ and it is limited by the V_1, V_2 values, consequently. Denote by $\eta(t)$ the number of customers present in the system at time instant t.

Let the volume vector of the arriving at time instant τ customer is equal (x_1, x_2) such that $\sigma_1(\tau^-) + x_1 \le V_1$, $\sigma_2(\tau^-) + x_2 \le V_2$ and $\eta(\tau^-) < n + m$. In this case, the customer is accepted to the system and we have $\eta(\tau) = \eta(\tau^-) + 1$, $\sigma_1(\tau) = \sigma_1(\tau^-) + x_1$, $\sigma_2(\tau) = \sigma_2(\tau^-) + x_2$. In opposite case, the customer is lost

and we obtain $\eta(\tau) = \eta(\tau^-)$, $\sigma_1(\tau) = \sigma_1(\tau^-)$, $\sigma_2(\tau) = \sigma_2(\tau^-)$. Let t be an epoch of service termination of the customer with volume vector (y_1, y_2). Then, the customer leaves the system and we have $\eta(t) = \eta(t^-) - 1$, $\sigma_1(t) = \sigma_1(t^-) - y_1$, $\sigma_2(t) = \sigma_2(t^-) - y_2$.

If we compare processes $\eta(t)$ in the system under consideration and classical one, we notice that their behavior does not differ in service termination epochs, whereas, in arriving epochs, we must take into account the limitation of the sectors of summary volume. For this reason, the process $\eta(t)$ is not Markovian in the analyzed system.

Let $\zeta_1^1(t), \zeta_1^2(t), \ldots, \zeta_{\eta(t)}^1(t), \zeta_{\eta(t)}^2(t)$ be the volumes of the parts of customers numbered by $1, \ldots, \eta(t)$ according to the order of their arrival to the system.

The behavior of the analyzed system can be described by the following Markovian process:

$$\left(\eta(t), \zeta_1^1(t), \zeta_1^2(t), \ldots, \zeta_{\eta(t)}^1(t), \zeta_{\eta(t)}^2(t)\right). \tag{1}$$

Process (1) can be characterized by the following functions:

$$P_k(t) = \mathsf{P}\{\eta(t) = k\}, k = \overline{0, n+m}; \tag{2}$$

$$G_k(t, x, y) = \mathsf{P}\{\eta(t) = k, \sigma_1(t) < x, \sigma_2(t) < y\}, k = \overline{1, n+m}, \tag{3}$$

where

$$\sigma_1(t) = \sum_{i=1}^{\eta(t)} \zeta_i^1(t), \sigma_2(t) = \sum_{i=1}^{\eta(t)} \zeta_i^2(t).$$

It is clear that, for every $k = \overline{1, n+m}$, we have:

$$P_k(t) = G_k(t, V_1, V_2). \tag{4}$$

In steady state that exists, if the condition $\rho = a/(n\mu) < \infty$ is satisfied, we can introduce the limits of the functions (2)–(3) when $t \to \infty$:

$$p_k = \mathsf{P}\{\eta = k\}, k = \overline{0, n+m}, \tag{5}$$

$$g_k(x, y) = \mathsf{P}\{\eta = k, \sigma_1 < x, \sigma_2 < y\}, k = \overline{1, n+m}, \tag{6}$$

where the random variables η, σ_1, σ_2 are the stationary analogies of the processes $\eta(t)$, $\sigma_1(t)$, $\sigma_2(t)$, consequently.

3 The Equations and Their Solution

By the process (1) analysis, we can write the following equations:

$$P_0'(t) = -aP_0(t)L(V_1, V_2) + \mu P_1(t); \tag{7}$$

$$P_1'(t) = aP_0(t)L(V_1, V_2) - a\int_0^{V_1}\int_0^{V_2} G_1(t, V_1 - x, V_2 - y)dL(x, y) - \mu P_1(t) + 2\mu P_2(t); \tag{8}$$

$$P_k'(t) = a\int_0^{V_1}\int_0^{V_2} G_{k-1}(t, V_1 - x, V_2 - y)dL(x, y)$$
$$-a\int_0^{V_1}\int_0^{V_2} G_k(t, V_1 - x, V_2 - y)dL(x, y) - k\mu P_k(t) + (k+1)\mu P_{k+1}(t), k = \overline{2, n-1}; \tag{9}$$

$$P_k'(t) = a\int_0^{V_1}\int_0^{V_2} G_{k-1}(t, V_1 - x, V_2 - y)dL(x, y)$$
$$-a\int_0^{V_1}\int_0^{V_2} G_k(t, V_1 - x, V_2 - y)dL(x, y) - n\mu P_k(t) + n\mu P_{k+1}(t), \tag{10}$$
$$k = \overline{n, n+m-1};$$

$$P_{n+m}'(t) = a\int_0^{V_1}\int_0^{V_2} G_{n+m-1}(t, V_1 - x, V_2 - y)dL(x, y) - n\mu P_{n+m}(t). \tag{11}$$

In steady state, from the equations (7)–(11), we obtain the following equations for the number p_0 and the functions (6):

$$0 = -ap_0L(V_1, V_2) + \mu p_1; \tag{12}$$

$$0 = ap_0L(V_1, V_2) - a\int_0^{V_1}\int_0^{V_2} g_1(V_1 - x, V_2 - y)dL(x, y) - \mu p_1 + 2\mu p_2; \tag{13}$$

$$0 = a\int_0^{V_1}\int_0^{V_2} g_{k-1}(V_1 - x, V_2 - y)dL(x, y) - a\int_0^{V_1}\int_0^{V_2} g_k(V_1 - x, V_2 - y)dL(x, y)$$
$$-k\mu p_k + (k+1)\mu p_{k+1},\ k = \overline{2, n-1}; \tag{14}$$

$$0 = a\int_0^{V_1}\int_0^{V_2} g_{k-1}(V_1 - x, V_2 - y)dL(x, y) - a\int_0^{V_1}\int_0^{V_2} g_k(V_1 - x, V_2 - y)dL(x, y)$$
$$-n\mu p_k + n\mu p_{k+1},\ k = \overline{n, n+m-1}; \tag{15}$$

$$0 = a \int_0^{V_1} \int_0^{V_2} g_{n+m-1}(V_1 - x, V_2 - y) dL(x, y) - n\mu p_{n+m}. \tag{16}$$

Denote by $L_k(x, y)$ the $k-$fold Stieltjes convolution of the distribution functions $L(x, y)$. More precisely we define it as follows:

$$L_0(x, y) \equiv 1, \; L_k(x, y) = \int_0^x \int_0^y L_{k-1}(x - u, y - v) dL(u, v).$$

In addition, we will use the following notation:

$$N(k) = \begin{cases} \frac{(n\rho)^k}{k!}, & \text{if } k = \overline{1, n}; \\ \frac{n^n \rho^k}{n!}, & \text{if } k = \overline{n+1, n+m}. \end{cases}$$

By direct substitution, we can find the solution of the equations (12)–(16) in the form:

$$g_k(x, y) = p_0 N(k) L_k(x, y), k = \overline{1, n+m}.$$

Then, using (4) (it is clear that in steady state we have $p_k = g_k(V_1, V_2)$), we can obtain the steady-state customers number probabilities p_k:

$$p_k = p_0 N(k) L_k(V_1, V_2), k = \overline{1, n+m}. \tag{17}$$

From the normalization condition, we finally obtain:

$$p_0 = \left[1 + \sum_{k=1}^{n+m} N(k) L_k(V_1, V_2) \right]^{-1}. \tag{18}$$

4 Loss Probability

The formula for the loss probability P_{loss} may be obtained using the equilibrium condition [2]:

$$a(1 - P_{loss}) = \mu \sum_{k=1}^{n-1} k p_k + n\mu (1 - \sum_{k=0}^{n-1} p_k).$$

The solution of the above equation leads to the following result:

$$P_{loss} = 1 - (n\rho)^{-1} \sum_{k=1}^{n-1} k p_k - \rho^{-1} (1 - \sum_{k=0}^{n-1} p_k), \tag{19}$$

where p_k are given by the relations (17).

5 The Main Statement

In this section, we will investigate the generalization (for the arbitrary l) of the model that was analyzed in the previous sections. First, we introduce some convenient notations for vectors:

$$\mathbf{x} = (x_1, \ldots, x_l);\ \mathbf{V} = (V_1, \ldots, V_l);\ \mathbf{u} = (u_1, \ldots, u_l);$$

$$\int_{(0,\mathbf{x})^l} f(\mathbf{x}-\mathbf{u})dL(\mathbf{u}) = \int_0^{x_1} \ldots \int_0^{x_l} f(x_1-u_1, \ldots, x_l-u_l)dL(u_1, \ldots, u_l).$$

The generalization of the queueing system $M/M/n/(m, V_1, V_2)$ is connected with the fact that we assume that every arriving customer is characterized by some l-dimensional random volume vector $\boldsymbol{\zeta} = (\zeta_1, \ldots, \zeta_l)$ and every sector of the summary volume $\boldsymbol{\sigma}(t) = (\sigma_1(t), \ldots, \sigma_l(t))$ is limited by values $V_1, \ldots, V_l$, consequently.

The above system can be denoted as $M/M/n/(m, \mathbf{V})$ and can be described by the following Markovian process:

$$\left(\eta(t), \boldsymbol{\zeta}_1(t), \ldots, \boldsymbol{\zeta}_{\eta(t)}(t)\right), \tag{20}$$

where $\boldsymbol{\zeta}_1(t), \ldots, \boldsymbol{\zeta}_{\eta(t)}(t)$ are the vectors of the volumes of the arriving customers numbered $1, \ldots, \eta(t)$ according to their arrival order, consequently.

Process (20) can be characterized by the following functions:

$$P_k(t) = \mathsf{P}\{\eta(t) = k\}, k = \overline{0, n+m}; \tag{21}$$

$$G_k(t, \mathbf{x}) = \mathsf{P}\{\eta(t) = k, \sigma_1(t) < x_1, \ldots, \sigma_l(t) < x_l\},\ k = \overline{1, n+m}, \tag{22}$$

where

$$\sigma_i(t) = \sum_{k=1}^{\eta(t)} \zeta_k^i(t),\ i = \overline{1, l}.$$

It is obvious that, for every $k = \overline{1, n+m}$, we have the following equality:

$$P_k(t) = G_k(t, \mathbf{V}). \tag{23}$$

In steady state, we can introduce the following functions that are the limits when $t \to \infty$ of the functions (21)–(22):

$$p_k = \mathsf{P}\{\eta = k\}, k = \overline{0, n+m}, \tag{24}$$

$$g_k(\mathbf{x}) = \mathsf{P}\{\eta = k, \sigma_1 < x_1, \ldots, \sigma_l < x_l\},\ k = \overline{1, n+m}, \tag{25}$$

where the random variables η, $\sigma_1, \ldots, \sigma_l$ are the stationary analogies of the processes $\eta(t), \sigma_1(t), \ldots, \sigma_l(t)$, consequently.

If we analyze the process (20), we can write down the following equations for the functions (21), (22):

$$P_0'(t) = -aP_0(t)L(\mathbf{V}) + \mu P_1(t); \tag{26}$$

$$P_1'(t) = aP_0(t)L(\mathbf{V}) - a\int_{(0,\mathbf{V})^l} G_1(t, \mathbf{V}-\mathbf{x})dL(\mathbf{x}) - \mu P_1(t) + 2\mu P_2(t); \tag{27}$$

$$P_k'(t) = a\int_{(0,\mathbf{V})^l} G_{k-1}(t, \mathbf{V}-\mathbf{x})dL(\mathbf{x}) - a\int_{(0,\mathbf{V})^l} G_k(t, \mathbf{V}-\mathbf{x})dL(\mathbf{x}) - k\mu P_k(t) + (k+1)\mu P_{k+1}(t),\ k = \overline{2, n-1}; \tag{28}$$

$$P_k'(t) = a\int_{(0,\mathbf{V})^l} G_{k-1}(t, \mathbf{V}-\mathbf{x})dL(\mathbf{x}) - a\int_{(0,\mathbf{V})^l} G_k(t, \mathbf{V}-\mathbf{x})dL(\mathbf{x}) - n\mu P_k(t) + n\mu P_{k+1}(t),\ k = \overline{n, n+m-1}; \tag{29}$$

$$P_{n+m}'(t) = a\int_{(0,\mathbf{V})^l} G_{n+m-1}(t, \mathbf{V}-\mathbf{x})dL(\mathbf{x}) - n\mu P_{n+m}(t). \tag{30}$$

In the stationary mode, from (26)–(30) we easily obtain:

$$0 = -ap_0L(\mathbf{V}) + \mu p_1; \tag{31}$$

$$0 = ap_0L(\mathbf{V}) - a\int_{(0,\mathbf{V})^l} g_1(\mathbf{V}-\mathbf{x})dL(\mathbf{x}) - \mu p_1 + 2\mu p_2; \tag{32}$$

$$0 = a\int_{(0,\mathbf{V})^l} g_{k-1}(\mathbf{V}-\mathbf{x})dL(\mathbf{x}) - a\int_{(0,\mathbf{V})^l} g_k(\mathbf{V}-\mathbf{x})dL(\mathbf{x}) - k\mu p_k + (k+1)\mu p_{k+1},\ k = \overline{2, n-1}; \tag{33}$$

$$0 = a\int_{(0,\mathbf{V})^l} g_{k-1}(\mathbf{V}-\mathbf{x})dL(\mathbf{x}) - a\int_{(0,\mathbf{V})^l} g_k(\mathbf{V}-\mathbf{x})dL(\mathbf{x}) - n\mu p_k + n\mu p_{k+1},\ k = \overline{n, n+m-1}; \tag{34}$$

$$0 = a\int_{(0,\mathbf{V})^l} g_{n+m-1}(\mathbf{V}-\mathbf{x})dL(\mathbf{x}) - n\mu p_{n+m}. \tag{35}$$

Denote by $L_k(\mathbf{x})$ the k-fold Stieltjes convolution of the functions $L(\mathbf{x})$. More precisely:

$$L_0(\mathbf{x}) \equiv 1,\ L_k(\mathbf{x}) = \int_{(0,\mathbf{x})^l} L_{k-1}(\mathbf{x}-\mathbf{u})dL(\mathbf{u}),\ k = 1, 2, \ldots.$$

By direct substitution, we can check that the solution of the (31)–(35) equations has the following form:

$$g_k(\mathbf{x}) = p_0 N(k) L_k(\mathbf{x}), k = \overline{1, n+m}.$$

Then on the base of (23) (and analogous formula in the stationary mode: $p_k = g_k(\mathbf{V})$), we can find formulae for p_k:

$$p_k = p_0 N(k) L_k(\mathbf{V}), k = \overline{1, n+m}, \tag{36}$$

whereas we have from the normalization condition that

$$p_0 = \left[1 + \sum_{k=1}^{n+m} N(k) L_k(\mathbf{V})\right]^{-1}. \tag{37}$$

The loss probability can be computed using (19), where p_k are definited in (36).

6 Special Cases and Numerical Examples

In this section, we suppose that components of the random volume vector of the arriving customer are independent (as it is often noticed in practice [15]), i.e. the joint distribution of the above vector $L(\mathbf{x}) = L(x_1, \ldots, x_l)$ can be presented in the product form:

$$L(\mathbf{x}) = \prod_{i=1}^{l} A^i(x_i). \tag{38}$$

Then the convolution $L_k(\mathbf{x})$ can be presented in the following form:

$$L_k(\mathbf{x}) = \prod_{i=1}^{l} A_k^i(x_i), \tag{39}$$

where $A_k^i(x_i)$, $i = \overline{1, l}$ are the k-fold convolutions of the functions $A^i(x_i)$, consequently. Then formulae (36) and (37) take the form:

$$p_k = p_0 N(k) \prod_{i=1}^{l} A_k^i(V_i), k = \overline{1, n+m}, \tag{40}$$

$$p_0 = \left[1 + \sum_{k=1}^{n+m} N(k) \prod_{i=1}^{l} A_k^i(V_i)\right]^{-1}. \tag{41}$$

6.1 $M/M/n/(0,\mathbf{V})$ Queueing System

Assume now that there is no queue in the system (we analyze $M/M/n/(0,\mathbf{V})$ system) and all the components of the random volume vector are independent and exponentially distributed with parameters $f_1,\ldots,f_l$, consequently. Then, from the relations (40) and (41), we obtain the following formulae:

$$p_k = p_0 \frac{(n\rho)^k}{k!} \prod_{i=1}^{l} \left[1 - e^{-f_i V_i} \sum_{m=0}^{k-1} \frac{(f_i V_i)^m}{m!} \right], \quad k = \overline{1,n}, \tag{42}$$

$$p_0 = \left\{ 1 + \sum_{k=1}^{n} \frac{(n\rho)^k}{k!} \prod_{i=1}^{l} \left[1 - e^{-f_i V_i} \sum_{m=0}^{k-1} \frac{(f_i V_i)^m}{m!} \right] \right\}^{-1}. \tag{43}$$

In addition, using (19), we obtain the following formula for the loss probability:

$$P_{loss} = 1 - (n\rho)^{-1} \sum_{k=1}^{n} k p_k, \tag{44}$$

where the probabilities p_k are definited by (42).

Now we illustrate our investigations with some numerical examples. Assume that we have $M/M/2/(0,\mathbf{V})$ queueing system in which every arriving customer is characterized by 3-dimensional random vector and its components are independent and exponentially distributed with parameters $f_1 = 1$, $f_2 = 2$ and $f_3 = 3$, consequently. The entrance Poisson flow parameter $a = 2$ and service time parameter $\mu = 5$. Then, from (42)–(44) we easily obtain:

$$p_1 = p_0 \prod_{i=1}^{3} \left[1 - e^{-iV_i} \right],$$

$$p_2 = \frac{p_0}{2} \prod_{i=1}^{3} \left[1 - e^{-iV_i} (1 + iV_i) \right],$$

$$p_0 = \left\{ 1 + \prod_{i=1}^{3} \left[1 - e^{-iV_i} \right] + \frac{1}{2} \prod_{i=1}^{3} \left[1 - e^{-iV_i} (1 + iV_i) \right] \right\}^{-1},$$

$$P_{loss} = 1 - p_1 - 2p_2.$$

The results of P_{loss} calculations for different values of V_1, V_2 and V_3 are presented in Table 1.

Table 1. Loss probability for the $M/M/2/(0,\mathbf{V})$ system

V_1	V_2	$P_{loss}(V_3 = 0,5)$	$P_{loss}(V_3 = 2,5)$	$P_{loss}(V_3 = 4,5)$
0,5	0,5	0,829984	0,783960	0,783796
0,5	2,5	0,741533	0,667462	0,667126
0,5	4,5	0,739375	0,664214	0,663870
2,5	0,5	0,642162	0,541505	0,540996
2,5	2,5	0,456468	0,292980	0,291996
2,5	4,5	0,450508	0,284285	0,283280
4,5	0,5	0,613546	0,501388	0,500791
4,5	2,5	0,408109	0,227011	0,225897
4,5	4,5	0,401261	0,217254	0,216118

6.2 $M/M/1/(\infty,\mathbf{V})$ Queueing System

Now we analyze the case of $M/M/1/(\infty,\mathbf{V})$ queueing system (it is obvious that our investigations are valid also for $m = \infty$). In this case, we have no limitation for the number of waiting customers but limited space vector causes the losses of the customers. Then we obtain the following formulae:

$$p_k = p_0 \rho^k \prod_{i=1}^{l} A_k^i(V_i), k \geq 1, \tag{45}$$

$$p_0 = \left[1 + \sum_{k=1}^{\infty} \rho^k \prod_{i=1}^{l} A_k^i(V_i) \right]^{-1}. \tag{46}$$

In the case when all the components of the random volume vector are independent and are exponentially distributed we obtain:

$$p_k = p_0 \rho^k \prod_{i=1}^{l} \left[1 - e^{-f_i V_i} \sum_{m=0}^{k-1} \frac{(f_i V_i)^m}{m!} \right], \ k \geq 1, \tag{47}$$

$$p_0 = \left\{ 1 + \sum_{k=1}^{\infty} \rho^k \prod_{i=1}^{l} \left[1 - e^{-f_i V_i} \sum_{m=0}^{k-1} \frac{(f_i V_i)^m}{m!} \right] \right\}^{-1}. \tag{48}$$

The loss probability can be calculated using the following formula [7]:

$$P_{loss} = 1 - \frac{1 - p_0}{\rho}. \tag{49}$$

The next numerical example is connected with the analysis of $M/M/1/(\infty,\mathbf{V})$ queueing system. The proper results were presented in the

Table 2. Loss probability for the $M/M/1/(\infty, \mathbf{V})$ system, $\rho = 0,5$

V_1	V_2	$P_{loss}(V_3 = 0,5)$	$P_{loss}(V_3 = 1,5)$	$P_{loss}(V_3 = 2,5)$
0,5	0,5	0,819382	0,771959	0,769339
0,5	1,5	0,733982	0,662921	0,658560
0,5	2,5	0,721641	0,646211	0,641417
1,5	0,5	0,661076	0,572484	0,566870
1,5	1,5	0,505325	0,369385	0,358726
1,5	2,5	0,480565	0,332841	0,320295
2,5	0,5	0,603727	0,498040	0,490775
2,5	1,5	0,419906	0,252940	0,237830
2,5	2,5	0,388834	0,204187	0,185520

Table 3. Loss probability for the $M/M/1/(\infty, \mathbf{V})$ system, $\rho = 3$

V_1	V_2	$P_{loss}(V_3 = 0,5)$	$P_{loss}(V_3 = 1,5)$	$P_{loss}(V_3 = 2,5)$
0,5	0,5	0,865001	0,836137	0,833961
0,5	1,5	0,817830	0,779984	0,776238
0,5	2,5	0,809875	0,769146	0,764645
1,5	0,5	0,790323	0,752991	0,749257
1,5	1,5	0,735557	0,698842	0,694337
1,5	2,5	0,725560	0,689149	0,684454
2,5	0,5	0,767154	0,728094	0,723685
2,5	1,5	0,712833	0,681753	0,678069
2,5	2,5	0,702940	0,675012	0,671976

formulae (47)–(49). Assume that arriving customers are characterized by 3-dimensional random volume vectors whose components are independent and exponentially distributed with parameters $f_1 = 1, f_2 = 2, f_3 = 3$, consequently. Using (48), we can obtain the approximations of the loss probabilities (we substitute the infinite sum appearing in (48) by the finite one – in this case $k = \overline{1, 12}$). The results of calculations when $\rho = \frac{1}{2}$ are presented in Table 2, and results for $\rho = 3$ are presented in Table 3.

7 Conclusions

In this paper, we investigate the modification of the classical $M/M/n/m$ queueing system, in which the arriving customers are characterized additionally by some random l-dimensional volume vector, and the total volume consists of l limited parts. In the analyzed model, service time and customers's random volume vector are independent. For this model, we obtain the steady-state distribution of the number of customers present in the system and loss probability formula as well. We also present the analysis of some special cases in which the components of the random volume vector are independent. We obtain exact formulae for the $M/M/n/(0, \mathbf{V})$ and $M/M/1/(\infty, \mathbf{V})$ queueing systems and illustrate them with some numerical examples in the case when components of the random volume vector are exponentially distributed.

References

1. Tikhonenko, O.M.: Queueing Models in Computer Systems. Universitetskoe, Minsk (1990) (in Russian)
2. Tikhonenko, O.: Computer Systems Probability Analysis. Akademicka Oficyna Wydawnicza EXIT, Warsaw (2006). (in Polish)
3. Alexandrov, A.M., Katz, B.A.: Non-homogeneous demands flows service. Izvestiya AN SSSR. Tekhnicheskaya Kibernetika **2**, 47–53 (1973). (in Russian)
4. Tikhonenko, O.M.: Destricted capacity queueing systems: determination of their characteristics. Autom. Remote Control **58**(6), 969–972 (1997)
5. Tikhonenko, O.M., Klimovich, K.G.: Queuing systems for random-length arrivals with limited cumulative volume. Prob. Inf. Transm. **37**(1), 70–79 (2001). https://doi.org/10.1023/A:1010451827648
6. Tikhonenko, O.M.: Generalized Erlang problem for service systems with finite total capacity. Prob. Inf. Transm **41**(3), 243–253 (2005). https://doi.org/10.1007/s11122-005-0029-z
7. Morozov, E., Nekrasova, R., Potakhina, L., Tikhonenko, O.: Asymptotic analysis of queueing systems with finite buffer space. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2014. CCIS, vol. 431, pp. 223–232. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07941-7_23
8. Tikhonenko, O.M.: Queuing systems with processor sharing and limited resources. Autom. Remote Control **71**(5), 803–815 (2010). https://doi.org/10.1134/S0005117910050073
9. Tikhonenko, O.: Queueing systems with common buffer: a theoretical treatment. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2011. CCIS, vol. 160, pp. 61–69. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21771-5_8
10. Yashkov, S.F., Yashkova, A.S.: Processor sharing: a survey of the mathematical theory. Autom. Remote Control **68**(9), 1662–1731 (2007). https://doi.org/10.1134/S0005117907090202
11. Bocharov, P.P., D'Apice, C., Pechinkin, A.V., Salerno, S.: Queueing Theory. VSP, Utrecht-Boston (2004)
12. Sengupta, B: The spatial requirements of an M/G/1 queue, or: how to design for buffer space. In: Baccelli, F., Fayolle, G. (eds.) Modelling and Performance Evaluation Methodology. LNCIS, vol. 60, pp. 547–562. Springer, Heidelberg (1984). https://doi.org/10.1007/BFb0005191

13. Schwartz, M.: Computer-communication Network Design and Analysis. Prentice-Hall, Englewood Cliffs, New York (1977)
14. Schwartz, M.: Telecommunication Networks: Protocols, Modeling and Analysis. Addison-Wesley Publishing Company, New York (1987)
15. Kim, H.-K.: System and method for processing multimedia packets for a network. US Patent No. 7,236,481 B2 (2002). https://patents.google.com/patent/US7236481B2/en
16. Chen, X., Stidwell, A.G., Harris, M.B.: Radio telecommunications apparatus and method for communications internet data packets containing different types of data. US Patent No. 7,558,240 B2 (2009). https://patents.google.com/patent/US7558240B2/en

GPU Accelerated Non-integer Order $PI^{\alpha}D^{\beta}$ Controller Used as AQM Mechanism

Adam Domański[1(✉)], Joanna Domańska[2], Tadeusz Czachórski[2], Jerzy Klamka[2], Dariusz Marek[1], and Jakub Szyguła[1]

[1] Institute of Informatics, Silesian University of Technology, Akademicka 16, 44-100 Gliwice, Poland
{adomanski,dariusz.marek,jakub.szygula}@polsl.pl
[2] Institute of Theoretical and Applied Informatics, Polish Academy of Sciences, ul. Bałtycka 5, 44-100 Gliwice, Poland
{joanna,tadek,jerzy.klamka}@iitis.pl

Abstract. The goal of this article was to reduce the time of dropping packed decisions for Active Queue Mechanisms based on $PI^{\alpha}D^{\beta}$. This article introduces a new way of performing calculations in a non-integer order controller using CUDA technology.

Keywords: Active Queue Management · Non-integer order $PI^{\alpha}D^{\beta}$ CUDA · GPU · Packet loss · Network congestion control

1 Introduction

Nowadays, the congestion problems are common in communication networks. The network routers send packets via links with limited bandwidth. If the number of incoming packets exceeds the capacity of a link, it becomes congested and the buffers of routers, that store packets, may overflow.

To solve this kind of problem in TCP/IP networks, the IETF organization (Internet Engineering Task Force), which develops and promotes Internet standards, recommends the use of Active Queue Management (AQM) mechanisms in IP routers. These mechanisms proactively drop packets, even if there is a place to store them, in order to announce that the router queue is increasing and there is a risk of congestion. The probability of packet rejection increases with the level of congestion. Due to the fact that packets are dropped randomly, only some users are notified and the phenomenon of global synchronization is therefore avoided. The mechanisms of active queue management enhance the efficiency of transmissions and cooperate with the congestion window mechanism of TCP protocol in adjusting the intensity of the stream to the level of network congestion.

First AQM mechanism was proposed in 1993 by Floyd and Jacobson. Its performance is based on a drop function giving probability that a packet is

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 286–299, 2018.
https://doi.org/10.1007/978-3-319-92459-5_23

rejected. It was named Random Early Detection (sometimes Random Early Discard). The probability of packet rejection depends on the moving (weighted) average of packets queue length determined at the moment of a new packet arrival. The moving average is determined according to the actual queue length and the previous (computed upon arrival of the previous packet) moving average. This approach creates a relationship between the probability of packet loss and the queue length in the previous time moments.

The RED mechanism is very useful in maintaining a reasonable level of average queue length and queue delay. However, the selection of the parameters is not a trivial issue [24,25]. If the parameters are not selected correctly, the performance of RED mechanism may deteriorate and then the TCP/RED system may become unstable [26]. The overview of the numerous studies on enhancing the performance of the primary RED mechanism can be found e.g. in work [15]. The analysis of the weighted average queue length implementation described in our paper [6] showed that the more detailed study of previous queue occupancy can improve the queue performance.

The AQM mechanism may be treated as a part of TCP/IP traffic control closed-loop. The feedback control system includes: (1) the TCP congestion window, which increases or decreases TCP stream as a function of losses, (2) packet buffer in a congested router, which reacts to the input stream changes, and (3) AQM mechanism, which defines the probability of the losses, which in turn, with certain delay, changes the size of congestion window.

The traditional mechanism used in automatics in feedback control system is a PI controller (Proportional- Integral controller). The extension of a PI controller is a PID controller (Proportional-Integral-Derivative controller). The PID controller is the mostly used controller in automatics, because of its simplicity, speed and availability of many efficient and simple methods of its adjusting, which in general requires the knowledge of the mathematical model of the object.

Based on the easy implementation of PI AQM controllers in real networks, a number of PI controllers have been proposed [18,26,27].

Theoretical studies have shown that in many cases the algorithms that use the fractional order differential-integral calculus work much better than their equivalents, based on the classical differential-integral calculus. Controllers of variable, fractional order ($PI^{\alpha}D^{\beta}$) are the generalization of the classical PID controllers. The first application of the fractional order PI controller as a AQM policy in fluid flow model of a TCP connection was presented in [17].

Our papers [7–9] describe how to use the response from PI^{α}(non-integer integral order)D^{β}(non-iterger derivative order) to calculate the probability of packet loss. This probability is described by a formula:

$$p_i = max\{0, -(K_P e_k + K_I \Delta^{\alpha} e_k + K_D \Delta^{\beta} e_k)\} \tag{1}$$

where K_P, K_I, K_D are tuning parameters, e_k is the error in current moment k : the difference between desired and actual queue length, and α and β are non integer integral and derivative orders.

The Fractional Order Derivatives and Integrals (FOD/FOI) definitions unify the definition of derivative and integral to one differintegral definition. The most

popular formulas to calculate differintegral numerically are Grunwald-Letnikov (GrLET) formula and Riemann-Liouville formulas (RL) [5,19,21].

Differintegral is a combined differentiation/integration operator. The q-differintegral of f, denoted by

$$\Delta^q f \tag{2}$$

is the fractional derivative (for $q > 0$) or fractional integral (if $q < 0$). If $q = 0$, then the q-th differintegral of a function is the function itself.

If order of differintegral is greater then 0 then we calculate the derivative. If the order is smaller then 0 then we calculate integral.

In the case of discrete systems (in the active queue management, packet drop probabilities are determined at discrete moments of packet arrivals) there is only one definition of differ-integrals of non-integer order. This definition is a generalization of the traditional definition of the difference of integer order to the noninteger order and it is analogous to a generalization used in Grunwald-Letnikov (GrLET) formula.

For a given sequence $f_0, f_1, ..., f_j, ..., f_k$

$$\triangle^q f_k = \sum_{j=0}^{k} (-1)^j \binom{q}{j} f_{k-j} \tag{3}$$

where $q \in R$ is generally a non-integer fractional order, f_k is a differentiated discrete function and $\binom{q}{j}$ is generalized Newton symbol defined as follows:

$$\binom{q}{j} = \begin{cases} 1 & \text{for } j = 0 \\ \dfrac{q(q-1)(q-2)..(q-j+1)}{j!} & \text{for } j = 1, 2, \ldots \end{cases} \tag{4}$$

Articles [7–9] show that using the non-integer order $PI^\alpha D^\beta$ controller as AQM mechanism has advantages over the standard RED mechanism and improves the router performance. However, the disadvantage of this solution is the increase in computational complexity of the algorithm to determine packet loss probability. The standard RED algorithm has a constant computational complexity. This complexity does not depend on the queue behavior and mechanism parameters. The non-integer order $PI^\alpha D^\beta$ algorithm has a computational complexity of the order k and this creates an increase in response time of the controller.

Experiments described in [7–9] have shown that the calculation time of the dropping packet probability is several times longer for PID controller compared to RED.

RED is one of the simplest AQM mechanism. The modifications of RED mechanism are often more complex. Some of them allow the adaptive setting of AQM parameters (e.g. ARED [13]). The adaptive mechanisms require the complicated calculations. Some mechanisms have involved the reduction in the bandwidth of aggressive streams. These mechanisms require a multiple packet draws from the queue (CHOKe algorithm family [4,11,22]) and sometimes the

construction of complex structures for storing auxiliary data [12]. The problem of computational complexity of these algorithms is significant especially for broadband computer networks. A delayed decision about a possible packet loss can cause a significant reduction of the transmission speed. There are articles making an attempt to improve the efficiency of AQM mechanisms (i.e. [2,20]). However, they only describe the performance of the network transmission completely ignoring the problem of delays caused by the AQM mechanisms in themselves.

The goal of this article was to reduce the dropping packed decisions time by (i) the decrease of the number k of elements in the sum in Eq. (3) by removing from the sum the oldest queue lengths, and (ii) use the parallel GPU architecture to shorten the calculation time.

To the best of our knowledge, there are no analyses of the GPU accelerated AQM implementation.

This article describes the influence of GPU architecture on the calculation time of the probability of packet loss in $PI^{\alpha}D^{\beta}$ controller. We implemented our algorithm on a software-based router and compare obtained response time on three types of GPU and CPU.

The remainder of the paper is organized as follows: Sect. 2 shortly describes the CUDA technology and the advantages of GPU based calculations. The section gives also some examples of algorithms implemented on GPU. Section 3 describes the GPU accelerated $PI^{\alpha}D^{\beta}$ controller and discusses obtained results. Some conclusions are presented in Sect. 4.

2 CUDA Technology

NVidia's Computed Unified Device Architecture (CUDA) is a parallel computing platform which allows to use GPU for time optimization of the code. CUDA gives a programmer a tool to use of parallelism and simple program mapping to the GPU code. CUDA is designed to solve problems that can be divided into tens, hundreds and thousands of parallel calculations.

During last few years the calculations on the GPU has been used in many scientific fields. There are many examples confirming the effectiveness of the CUDA implementation, presented e.g. in [10,16]. Paper [28] shows the practical construction of a system using the GPU to ensure the realtime operation. GPU offers many more implementation choices. Several effective techniques for implementation of vectors and vector-matrix multiplication in CUDA were investigated e.g. using single and double precision or using GPU cache memory and calculation without cache memory [3]. The article [16] shows the programmer tools and technics based on high-level languages.

Article [14] shows that the transfer of data between the CPU and the GPU introduces delays. In this transfer we need to copy the entire block of memory from the CPU to the GPU. The high-level language Python uses a list to store the block of memory. In the case of lists, data block can be placed in different parts of memory. Hence, before sending to GPU, data must be stored in memory as an array (continuous memory block). Multiple lists into arrays conversions may cause additional delays [23].

Our paper [10] described the GPU implementation of the Fluid-Flow approximation method. This implementation used the Python language. Access to nVidia's CUDA parallel computation API [1] was obtained using PyCUDA. The NumPy (library) model allows us to use the arrays. Our results presented in Sect. 3 are obtained with the use of the mentioned above tools. The Listing 1 presents the integral (derivative) calculation in CUDA.

```
# CUDA calculating module.
# Computation the sum of elements in the array
# CUDA calculating module.
# Multiplication the coefficients and queue occupancy history
import pycuda.autoinit
import pycuda.driver as drv
from pycuda.compiler import SourceModule
import pycuda.gpuarray as gpuarray
import numpy as np

#CUDA modules

# CUDA calculating module.
# Multiplication the coefficients and queue occupancy history
# dest - results array
# queueHistory
# coefficients
# n-numer of multiplication(size of queueHistory or coefficients)

multiplyModule = SourceModule("""
     __global__ void multiplyKernel
     (float *dest, float * queueHistory,
     float  * coefficients, int * n)
    {
        //calculate index of element in array
        const int i = blockDim.x * blockIdx.x + threadIdx.x;
        if(i < n[0])
        {
            //multiply element in queue history by coefficient
            dest[i] = queueHistory[i] * coefficients[i];
        }
    }
    """)

multiplyKernel = multiplyModule.get_function("multiplyKernel")


# CUDA calculating module.
# Computation the sum of elements in the array
```

```
# dest - results array
# vectorToSum - array with data to sum
# n - numer of additions (size of vectorToSum)

additionModule = SourceModule("""
    __global__ void additionKernel
   (float *dest, float * arrayToSum, int * n)
   {
       //calculate index of element in array
       const int i = blockDim.x * blockIdx.x + threadIdx.x;
       if(i < n[0])
        {
//CUDA array addition core function - sum of array elements
            atomicAdd(dest, arrayToSum[i]);
        }
    }
                                  """)
additionKernel = additionModule.get_function("additionKernel")

coefficientsVector = np.array()
queueOccupancyHistory = np.array()

blocksize = len(queueOccupancyHistory)
gridsize = 1

# create result array for CUDA multiplications

occupancyHistoryMultipyByCoefficientVector
   = np.zeros(len(queueOccupancyHistory ))

# calculate number of CUDA block and CUDA grid size

if blocksize > multiplyKernel.max_threads_per_block:
    blocksize = multiplyKernel.max_threads_per_block
    gridsize = int(math.ceil(len(queueOccupancyHistory )
                    /float(blocksize)))

# store queue occupancy history length in numpy array

queueOccupancyHistoryLength
   = np.array([len(queueOccupancyHistory)])

# send data to CUDA and run multiplications on CUDA
# data:
# occupancyHistoryMultipyByCoefficientVector - results array
# queueOccupancyHistory
```

```
# coefficientsVector
# queueOccupancyHistoryLength - number of computations

multiplyKernel(
    drv.Out(occupancyHistoryMultipyByCoefficientVector),
      drv.In(queueOccupancyHistory),
      coefficientsVector, drv.In(queueOccupancyHistoryLength ),
    block=(blocksize,1,1), grid=(gridsize,1))

# create result array for CUDA additions
result = np.array([0])

# send data to CUDA and run computations (additions in CUDA)
# data:
# result - results array
# occupancyHistoryMultipyByCoefficientVector
# - data get from previous CUDA computations
# queueOccupancyHistoryLength - number of computations

additionKernel(
    drv.Out(result),
    drv.In(occupancyHistoryMultipyByCoefficientVector),
    drv.In(queueOccupancyHistoryLength ),
    block=(blocksize,1,1), grid=(gridsize,1))

#variable result contains calculated I or D term
```

Listing 1. Implementation of the coefficient multiplication and adding vector of results in CUDA

3 The Packet Dropping Probability Function (Based on $PI^{\alpha}D^{\beta}$ Controller) Accelerated on GPU

A simple iterative implementation of $PI^{\alpha}D^{\beta}$ controller results in increase of delays in controller response. These delays may result in the lower performance of network transmissions. Our main goal is the acceleration of the controller response. The redaction of calculation time is possible in two ways: reducing the number of PID calculations and using the parallel GPU architecture.

Our work consisted of the implementation of the non-integer order $PI^{\alpha}D^{\beta}$ controller and testing the algorithm in the real software router. In experiments we changed the architecture of the computers. In the research we used 3 different GPU architectures:

1. NVIDIA GeForce GT 540M
2. NVIDIA GeForce GTX 1060
3. NVIDIA Quadro K2000

We also consider 3 different computer configurations:

1. Intel Core i5-3570 CPU @ 3.40 GHz
2. Intel Core i7-2670QM CPU @ 2.20 GHz
3. Intel Core i5-7300HQ CPU @ 2.50 GHz

We also changed the calculations by removing the oldest events from the history of the queue, i.e. limiting the horizon of differentiation/integration operator.

We focus on the relationship between the number of events in historical queue occupancy (value k, see Eq. 3), computer hardware (especially the GPU) and response time of controller. The response time of the non-integer order $PI^{\alpha}D^{\beta}$ controller is a necessary time to complete the controller calculations (see Eq. 1).

Figure 1 presents the $PI^{\alpha}D^{\beta}$ controller response time according to the number of events in queue occupancy history considered in the calculations. Obtained results meet our expectations. The worst results were obtained in the case of the computer with an old generation i7 processor (4 cores, 8 threads, base clock 2.3, boost clock 3.10 GHz). The better results were obtained for computers with i5 processors. These results were similar because of the same processor specifications (4 cores, boost clocks between 3.5 and 3.8 GHz).

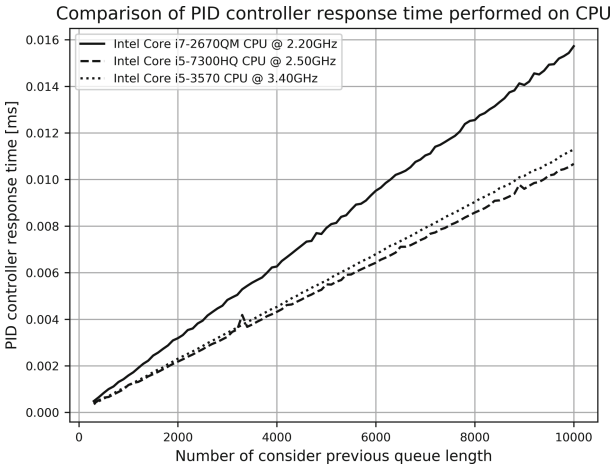


Fig. 1. Comparison of $PI^{\alpha}D^{\beta}$ controller response time performed on CPU

In all of these cases increase in the number of terms of the sum in $PI^{\alpha}D^{\beta}$ probability calculations brings to increase in response time of the $PI^{\alpha}D^{\beta}$ controller. For a large number of terms, long response time may affect the incorrect (to slow) action of the controller. This problem can be solved by using GPU.

Figure 2 presents the same relationship as Fig. 1 but calculations (multiplication and addition) were performed on the GPU (as a comparison we add one CPU result). These results show that the response time of the $PI^{\alpha}D^{\beta}$ controller does

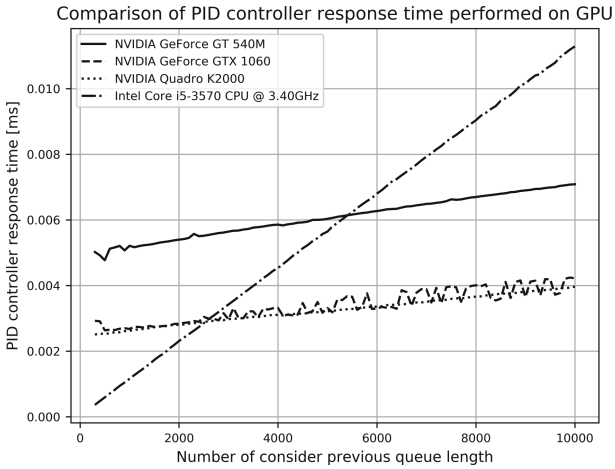


Fig. 2. Comparison of $PI^{\alpha}D^{\beta}$ controller response time performed on GPU

not increase as fast as on CPU. Obtained results show that the GPU implementation of the controller demonstrates the advantage when the size of the queue occupancy history exceeds (depends on GPU model) about 2000 items.

GPU introduces an additional delay associated with the transfer data between the CPU and the GPU. Figure 3 presents the relationship between the transfer time and the number of sent data. The transfer time is small and does not strongly depend on computer architecture and type of GPU. The time of transfer between GPU and CPU is about 12% of the total calculation time and does not depend significantly on the size of the transmitted data. Table 1 presents more detailed results.

Table 1. The times of the transfer between CPU and GPU

Nb of terms	Times [ms]		[%]
	Transfer between CPU and GPU	PID response	
1000	3.05400E−4	2.47440E−3	12.34
2000	3.05399E−4	2.52970E−3	12.07
3000	3.10400E−4	2.52109E−3	12.31
4000	3.21099E−4	2.55230E−3	12.58
5000	2.98499E−4	2.53129E−3	11.79
6000	2.81999E−4	2.54939E−3	11.06
7000	3.17300E−4	2.57530E−3	12.32
8000	3.07000E−4	2.58740E−3	11.86
9000	3.10399E−4	2.60139E−3	11.93
10000	3.22699E−4	2.59440E−3	12.43

Due to this technological delay it is not favourable to use GPU for short sequences. For long sequences, the advantages of GPU are more visible.

A data which are transferred to the GPU must have a form of an array (see Sect. 2). Regular sets of data are usually stored (in Python) in the form of lists and tuples. Therefore, each list has to be converted before transferring to the GPU. This conversion unnecessarily increases the response time of the controller. Moreover, the increase in the size of converted data results in conversion's time increase (see Table 2). Fortunately, additional Python modules (i.e. NumPy) allow us to use arrays. This modified approach (using arrays instead of lists) is a bit more complex, but it allows to avoid the unnecessary conversions. The complexity of this approach is a consequence of difficulties in selection of elements to be transferred. The results described below demonstrate that in spite of its complexity the proposed approach gives better results than the method using lists into arrays conversions.

Table 2. The times of the conversion from list to array

Nb of terms	Times [ms]		[%]
	Conversion list to array	PID response	
1000	8.59000E−05	2.47440E−03	3.47155
2000	1.67200E−04	2.52970E−03	6.60948
3000	2.50000E−04	2.52109E−03	9.91635
4000	3.32800E−04	2.55230E−03	13.03922
5000	4.15700E−04	2.53129E−03	16.42246
6000	4.96900E−04	2.54939E−03	19.49094
7000	5.81200E−04	2.57530E−03	22.56824
8000	6.64100E−04	2.58740E−03	25.66669
9000	7.47000E−04	2.60139E−03	28.71542
10000	8.29700E−04	2.59440E−03	31.98042

Figures 4 and 5 present the influence of proposed approach on the response time of the $PI^{\alpha}D^{\beta}$ controller (GPU and CPU implementations). These results confirm that our algorithm reduces the response time of the $PI^{\alpha}D^{\beta}$ controller for calculations performed on the GPU. It was to be expected that this approach increases the time for calculations performed on the CPU.

Figure 6 presents the comparison of the response times of the $PI^{\alpha}D^{\beta}$ controller obtained for two data format (float32 and float64). It was to be expected that the data format does not significantly influence the controller response time.

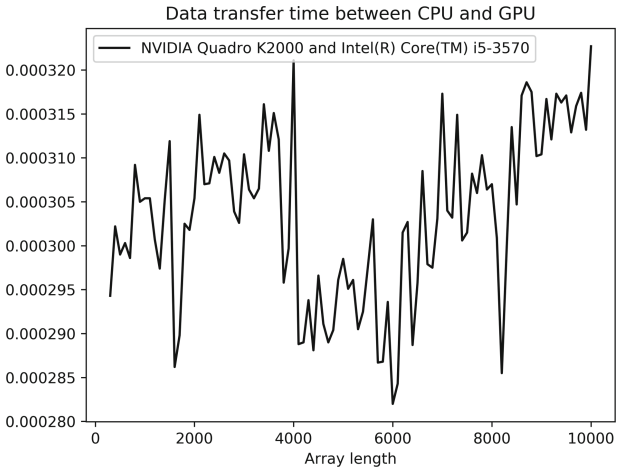


Fig. 3. Transfer between CPU and GPU times [ms]

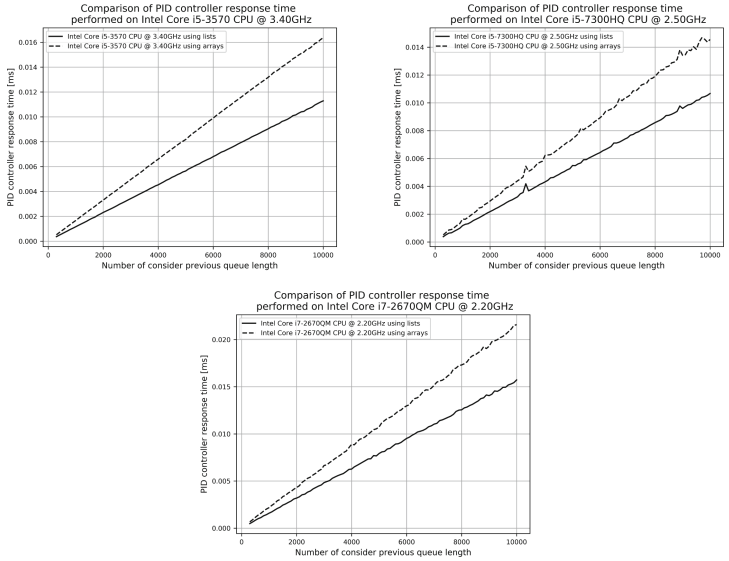


Fig. 4. Comparison of $PI^{\alpha}D^{\beta}$ controller response time performed on CPUs with implemented arrays and lists

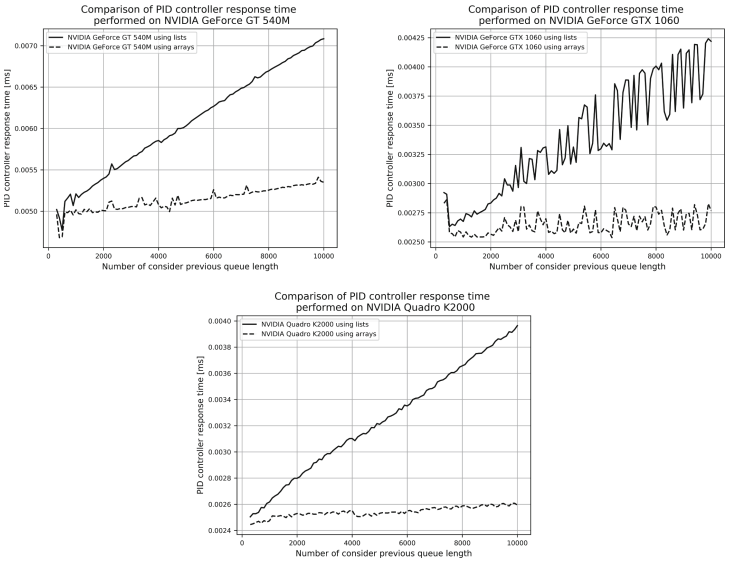


Fig. 5. Comparison of $PI^{\alpha}D^{\beta}$ controller response time performed on GPUs with implemented arrays and lists

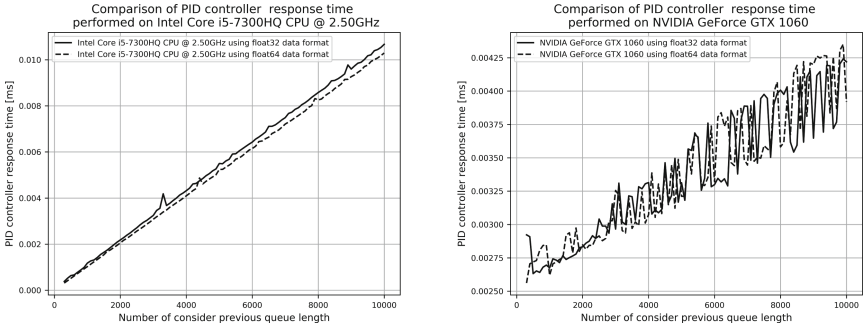


Fig. 6. Comparison of $PI^{\alpha}D^{\beta}$ controller response time performed on GPUs and CPUs with float32 and float64 data format

4 Summary

The Internet Engineering Task Force (IETF) organization recommends that IP routers should use the active queue management mechanism (AQM). The basic algorithm for AQM is the RED algorithm. There are many modifications and improvements of the RED mechanism. One of these improvements is the calculation of the probability of packet loss using a $PI^{\alpha}D^{\beta}$ controller. Our previous

work has shown the advantage of this solution. Unfortunately, this process is time-consuming.

Our paper introduces a new way of non-integer order $PI^{\alpha}D^{\beta}$ controller calculations - using CUDA technology.

In our experiments we calculate the response time of the controller. This response time is used to calculate the packet loss probability. The results obtained using the GPU architecture were compared to the time of calculation obtained using the CPU architecture. We used three different CPU and three GPU architectures. Our results confirm that using GPU shorten the response time of a $PI^{\alpha}D^{\beta}$ controller.

The second aspect discussed in our paper is the number of the sum terms taken into account during non-integer integral or non-integer derivative calculations. In the case of decrease in the number of terms, we don't take into account the oldest queue lengths. These research confirms that the use of GPU is efficient for long sequences of historical queue lengths. In our future work we will try to evaluate the influence of the number of terms k on $PI^{\alpha}D^{\beta}$ controller. We expect that it depends on the nature of incoming traffic, namely on the degree of its self-similarity and on its long term autocorrelation.

We believe that the presented here results confirm the benefits of the utilisation of the $PI^{\alpha}D^{\beta}$ controller.

References

1. CUDA programming guide (2012)
2. Al-Faiz, M.Z., Sabry, S.S.: Optimal linear quadratic controller based on genetic algorithm for TCP/AQM router. In: International Conference on Future Communication Networks (2012)
3. Bell, N., Garland., M.: Implementing sparse matrix-vector multiplication on throughput-oriented processors. In: Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis (2009)
4. Chhabra, P., Chuig, S., Goel, A., John, A., Kumar, A., Saran, H., Shorey, R.: Xchoke: malicious source control for congestion avoidance at internet gateways. In: 10th IEEE International Conference on Network Protocols (2002)
5. Leszczyński, J., Ciesielski, M.: A numerical method for solution of ordinary differential equations of fractional order. In: Wyrzykowski, R., Dongarra, J., Paprzycki, M., Waśniewski, J. (eds.) PPAM 2001. LNCS, vol. 2328, pp. 695–702. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-48086-2_77
6. Domańska, J., Domański, A., Augustyn, D., J. Klamka: A RED modified weighted moving average for soft real-time application. Int. J. Appl. Math. Comput. Sci. **24**(3), 697–707 (2014)
7. Domańska, J., Domański, A., Czachórski, T., Klamka, J.: Use of a non integer order PI controller to active queue management mechanism. Int. J. Appl. Math. Comput. Sci. **26** (2016)
8. Domański, A., Domańska, J., Czachórski, T., Klamka, J.: Self-similarity traffic and AQM mechanism based on non-integer order $PI^{\alpha}D^{\beta}$ controller. In: Gaj, P., Kwiecień, A., Sawicki, M. (eds.) CN 2017. CCIS, vol. 718, pp. 336–350. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59767-6_27

9. Domański, A., Domańska, J., Czachórski, T., Klamka, J., Szyguła, J.: The AQM dropping packet probability function based on non-integer order $PI^{\alpha}D^{\beta}$ controller. In: Ostalczyk, P., Sankowski, D., Nowakowski, J. (eds.) RRNR 2017. LNEE, vol. 496, pp. 36–48. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-78458-8_4
10. Domański, A., Domańska, J., Czachórski, T.: GPU-accelerated fluid flow approximation of the Active Queues Management algorithms. Theor. Appl. Inf. (2013)
11. Eshete, A., Jiang, Y.: Generalizing the choke flow protection. Comput. Netw. **57**(1), 147–161 (2013)
12. Feng, W., Shin, K., Kandlur, D., Saha., D.: The BLUE active queue management algorithms. IEEE/ACM Trans. Netw. (2002)
13. Floyd, S., Gummadi, R., Shenker, S.: Adaptive RED: an algorithm for increasing the robustness of RED's active queue management. IEEE Trans. Control Syst. Technol. (2001)
14. Fujii, Y., Azumi, T., Nishio, N., Kato, S., Edahiro, M.: Data transfer matters for GPU computing. In: Parallel and Distributed Systems (ICPADS) (2013)
15. Hassan, M., Jain, R.: High Performance TCP/IP Networking. Pearson Education Inc. (2004)
16. Klöckner, A., Pinto, N., Lee, Y., Catanzaro, B., Ivanov, P., Fasih, A.: GPU run-time code generation for high-performance computing. Parallel Comput. (2009)
17. Krajewski, W., Viaro, U.: On robust fractional order PI controller for TCP packet flow. In: BOS Coference: Systems and Operational Research, Warsaw, Poland, September 2014
18. Michiels, W., Melchor-Aquilar, D., Niculescu, S.: Stability analysis of some classes of TCP/AQM networks. Int. J. Control **79**, 1136–1144 (2006)
19. Miller, K., Ross, B.: An introduction to the fractional calculus and fractional differential equations. Wiley, New York (1993)
20. Pibiri, G., Goldrick, C.M., Huggard, M.: Enhancing AQM performance on wireless networks. In: IFIP Wireless Days (2012)
21. Podlubny, I.: Fractional differential equations. Academic Press, San Diego (1999)
22. Pan, R., Balaji Prabhakar, K.P.: Estatles AQM scheme for approximating fair bandwidth allocation. IEEE INFOCOM (2000)
23. Sanders, J., Kandrot., E.: CUDA by Example: An Introduction to General-Purpose GPU Programming. Addison-Wesley Professional (2010). ISBN 0131387685
24. Sawicki, M., Kwiecień, A.: Unexpected anomalies of isochronous communication over USB 3.1 gen 1. Comput. Stand. Interfaces (2017)
25. Tan, L., Zhang, W., Peng, G., Chen, G.: Stability of TCP/RED systems in AQM routers. IEEE Trans. Autom. Control **51**(8), 1393–1398 (2006)
26. Unal, H., Melchor-Aguilar, D., Ustebay, D., Niculescu, S.I., Ozbay, H.: Comparison of PI controllers designed for the delay model of TCP/AQM. Comput. Commun. **36**, 1225–1234 (2013)
27. Ustebay, D., Ozbay, H.: Switching resilient pi controllers for active queue management of TCP flows. In: Proceedings of the 2007 IEEE International Conference on Networking, Sensing and Control, pp. 574–578 (2007)
28. Wodziński, M., Krzyżanowska, A.: Sequential Classification of Palm Gestures Based on Algorithm and MLP Neural Network for Quadrocopter Control. Metrol. Meas. Syst. (2017)

Consequences of the Form of Restrictions in Coloured Petri Net Models for Behaviour of Arrival Stream Generator Used in Performance Evaluation

Dariusz Rzonca(✉), Wojciech Rząsa, and Sławomir Samolej

Department of Computer and Control Engineering,
Rzeszow University of Technology,
Al. Powstancow Warszawy 12, 35-959 Rzeszow, Poland
{drzonca,wrzasa,ssamolej}@prz.edu.pl

Abstract. Coloured Petri Nets formalism is commonly used for modelling various aspects of communication systems. Simulations of such models facilitate performance evaluation. Typically during the simulation a stream of data (tokens) is produced in the part of the model called stream generator, as an input for the latter part of the model. In the paper different models of stream generators are discussed, as well as their influence on distribution of resulting tokens, during the simulation in the CPN Tools software.

Keywords: Coloured Petri Nets · Performance evaluation · Simulation

1 Introduction

Performance evaluation is a well-established branch of modern computation techniques. It's main task is to create computational or mathematical model of these systems where a flow of some units or fluids determine their dynamics. The units are somehow stored (or queued), serviced and distributed among branches of the system. This in consequence results in time delays observed during the units flow through the system.

The research reported in this paper is a part of long-term programme dedicated to successful application of Coloured Petri Nets (CPN) [11] for distributed systems modelling, development and performance evaluation. So far some principles and backgrounds of a new software tool as well as systematic methodology of modelling distributed systems was introduced [19,30]. Then the methodology was successfully applied for simulation evaluation of internet systems load balance modelling [32], computer cluster reconfiguration analysis [31], and priority-based internet request scheduling evaluation [33].

The other branch of the research includes performance evaluation of different types of distributed systems. Analysis of the systems is based on models created

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 300–310, 2018.
https://doi.org/10.1007/978-3-319-92459-5_24

using CPN formalism, however the CPN model is automatically created from a High-level model of analysed system as in [21]. This approach was used to evaluate performance of a Distributed Control System [10,24], explain performance issues connected with Platform as a Service designed to host web applications [23], and to support decision concerning scaling resources for a web application running on a distributed platform [26]. One of important tools used in this part of our research was original CPN simulator. The simulator was designed to enable integration between formalism-based simulation and object-oriented programming.

One of the recently explored branches of the research conducted by authors of this paper is arrival stream modelling and analysis. According to a distributed system modelling method introduced in [30,32] the system modelled has a structure similar to an open queuing system. Consequently, one of its intrinsic components is an arrival stream generator (comp. Fig. 1). It produces a stream of data which is distributed over the nodes of modelled system and finally send back to the system's environment. The delay between the time of arrival of data and the response of the system evoked by this data is one of the main benchmarks during the system performance evaluation. Additionally, the analysis of distribution of data packages within the system makes it possible to find out potential bottlenecks.

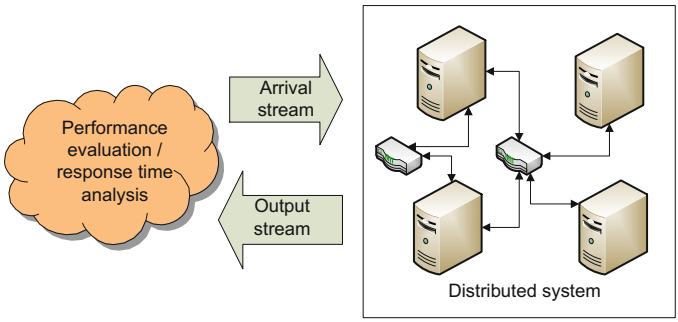


Fig. 1. Idea of response time performance evaluation

Typically, the arrival stream generator produces an input data stream in a random order according to predefined distribution [4,5,30,32,35]. In some research a recorded data stream is used as an input to the system model under performance evaluation [34]. Alternatively a new mathematical model of distribution of arriving data is proposed, such as in [13]. The particular aim of the research reported in this paper follows the last of above mentioned approaches.

We will discuss a specific behaviour of a sample arrival stream generator. Its variants should behave similarly, however we will reveal that they produce significantly different data streams depending on the way some Coloured Petri Net constraints are set. We will describe variants of the model and discuss it's

behavior. We will also describe tests performed with the TCPN Tools[1] – the most popular simulator of Timed Coloured Petri Nets. Finally, we will discuss observed results and explain them in the light of the TCPN definition and assumptions concerning the simulator. Although, the revealed behaviour of TCPN models is fully consistent with the TCPN definition, but it may be considered counter intuitive, consequently leading to considerable errors and incorrect conclusions drawn from simulating TCPN models.

We believe that results presented in this paper may influence CPN models of real applications, related to various classes of computer networks.

2 Related Work

Performance evaluation is systematically applied in computer and communication networks development [2,5,8,12,19,30,34], manufacturing systems assessment [3,9] as well as business processes analyses [6,14]. Two dominant mathematical formalisms currently used for solving performance evaluation and analysis problems are queuing nets [4,5] and Petri Nets [7,9]. Queuing nets make it possible to calculate the throughput of the subsequent elements of a computing (telecommunication, manufacturing, business, etc.) process represented as a network of queuing systems. Petri Nets are specific graphs where the token flow through the network can be interpreted as the reallocation of the load in the evaluated dynamic system. It is also worth noticing that there exist mathematical formalisms successfully merging queuing and Petri nets, called Queuing Petri Nets. Among other applications, Queuing Petri Nets [1] were successfully applied for distributed web systems modelling and performance evaluation [12,16].

In the previous editions of the International Science Conference on Computer Networks numerous papers were presented, where different aspects of communication in real-world applications were modelled and analysed using Petri nets. Target fields were very broad, from industrial networks and PLC controllers communication subsystems [24,27–29], wireless mesh networks [17], TCP-based networks [20,22,25], distributed web systems [18] to large-scale Platform as a Service cloud-based infrastructure [23].

3 Stream Generation Models

The Coloured Petri Nets (CPN) models of stream generators are typically used during a simulation to produce a set of tokens with necessary parameters. In some cases such a token represents mapping between two different sets. Typical example of such case is the assignment of some resources (nets, servers, CPUs, etc.) to users. Simple model of such generator is shown in Fig. 2.

Users place models users, whereas *Res* place models available resources. Initial marking on the places represents two users (*User1* and *User2*) and two resources (1 and 2). Arcs between these places and *Generator* transition are

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

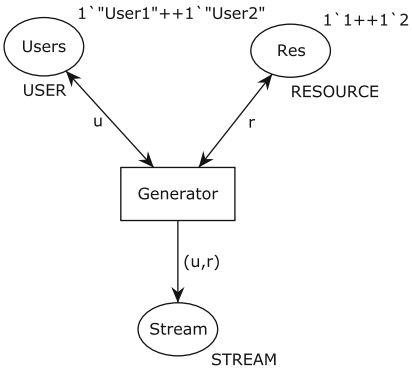


Fig. 2. Simple CPN model of stream generator

bidirectional, thus tokens representing user and resource used in some transition firing, immediately return to these places and are always available. More complicated models, with limited availability may be considered, but such a simplified model has been intentionally chosen here for brevity. Firing the *Generator* transition creates a token in the *Stream* place, assigns one of the resources to one of the users. In this simple model every binding of the arc variables to the tokens is allowed, modelling situation, where any user can use any resource in the stream generation process.

Of course such case, where mappings between users and resources are unrestricted, is the simplest, but unrealistic one. Let us analyse the following case: we would like to model the hypothetical network where resource 1 can be used only by *User1*, whereas resource 2 may be used by both users. The natural way to model such behaviour seems to include appropriate guard in the *Generator* transition, i.e.:

```
(u="User1" andalso r=1) orelse r=2
```

and create model presented in the Fig. 3.

This model allows to fire the *Generator* transition for any user if resource value equals 2. Resource number 1 can be used to fire the transition only if *User1* is used. This way specific binding is forced to enable generation of a *Stream* token.

Another way to model restrictions of the user/resource mapping involves appropriate inscription on the output arc. Such inscription should check if randomly chosen binding of the input variables during firing the transition fulfils the requirements. In the considered case the actual expression would be as follows:

```
if (u="User1" andalso r=1) orelse r=2 then 1‘(u,r) else empty
```

The complete model is presented in the Fig. 4.

In this model, if considered binding includes a *User1* token and value 1 representing resource, a *Stream* token is generated. Similarly if resource token is bound to value 2 (and user token bound to any value) a *Stream* token is generated. In other cases, marking put in the *Stream* place is empty, i.e. no

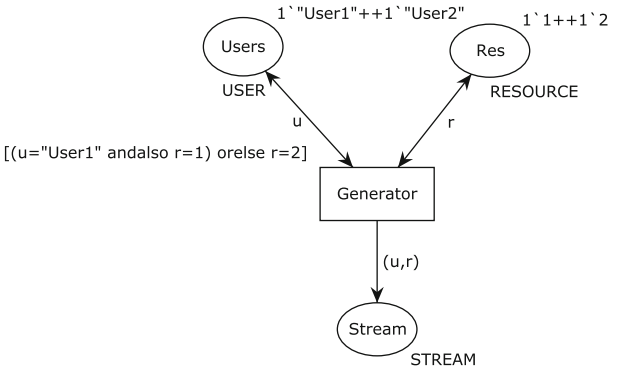


Fig. 3. Model of the stream generator with guard

token is generated. Thus, functionally this solution corresponds to the one from Fig. 3.

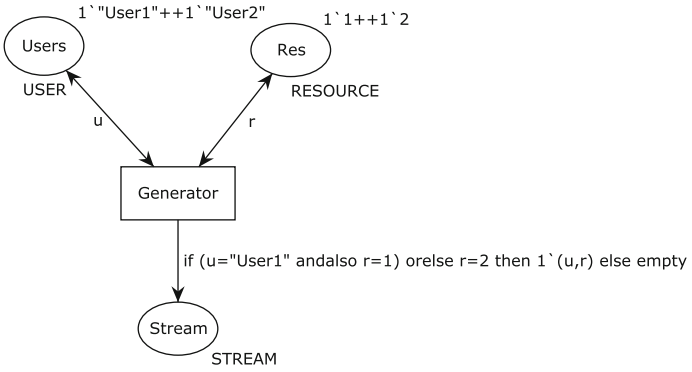


Fig. 4. Model of the stream generator with restriction on the output arc

4 Distribution of Generated Values

As discussed in Sect. 3 the two models presented in the Figs. 3 and 4 are functionally identical: they both ensure that in order to generate a stream, resource 1 can be used only by *User1* and resource 2 can be used by any user. However, if we consider simulation, another question arises: what distribution of the user/resource pairs in the *Stream* place should be generated? The answer is quite trivial for the basic model from the Fig. 2. Each pair of user/resource tokens is allowed, thus during simulation each token from each input place should be randomly chosen with equal probability, so simulation of such net should provide similar number of tokens for each user/resource pair. It is however more complicated for the other two models.

Both arcs in the models: between the *Users* place and the *Generator* transition and between the *Res* place and the *Generator* transition are two-way, thus the tokens from each of these places can be used arbitrary number of times to fire the *Generator* transition. The condition restricting pairs of user/resource values that can appear in the *Stream* place is asymmetric, preferring the *User1* over the *User2* giving the first one opportunity to use any resource. This may suggest, that there should be twice more tokens with *User1* generated in the output place. On the other hand, one may expect, that both tokens from *Users* place should be used equal number of times and thus, resource 2 should probably be used more frequently (since it can be used by any user). Consequently, the output place would have comparable number of tokens generated by each user and more tokens generated using resource 2 than tokens generated using resource 1.

The book describing Coloured Petri Nets [11] does not explain how the problem should be solved. Actually, it does not discuss the detailed problems concerning simulation of the CPN. This problem appeared while we were working on the CPN simulator mentioned before and used in our research.

5 Testing in CPN Tools

In order to decide what solution is proper for the problem, we first verified behaviour of discussed models in CPN Tools[2] – the software, that should probably be considered a reference implementation of a CPN simulator. The models were created in the CPN Tools editor and simulated. For every model the *Generator* transition was fired 100,000 times in order to produce reliable information about distribution of tokens used to enable the transition.

Simulation results for the first, basic example are shown in Fig. 5. In this model every pair of tokens from the two input places could be used to fire the transition. As expected after 100,000 firing of the *Generator* transition, approximately 25,000 tokens for each user/resource pairs were produced. Thus, we may conclude that pairs of tokens enabling the transition were used equally frequently.

Results of simulation for the model from Fig. 3 are presented in the Fig. 6. In this case guard is used to limit the valid pairs of user/resource tokens.

At first sight these results seem to be counter-intuitive. Resulting pairs do not have equal distribution any more. Simulator does not evenly select user/resource pairs from the whole space of enabling bindings. There are almost 50,000 tokens for the *User 1/resource 1* pair, and approximately 25,000 tokens for each *User 1/resource 2* and *User 2/resource 2* pairs. We may see that simulator uses each resource token equally frequently, and finds matching user for selected resource.

Simulation results for the last model, the one where arc expression is used to restrict user/resource matching, are presented in the Fig. 7. In this case it can be seen that there is equal distribution of the resulting pairs. In this case the result seems to prove that the simulator evenly selects each value from the space

[2] http://cpntools.org/.

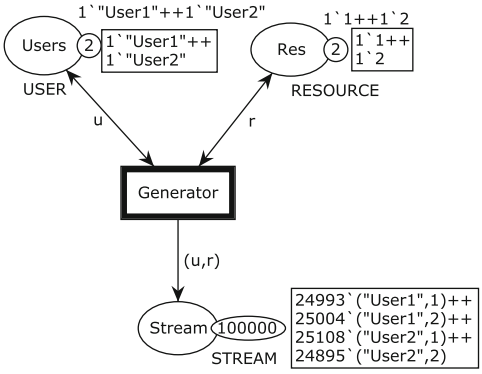


Fig. 5. Simulation results for the simple stream generator

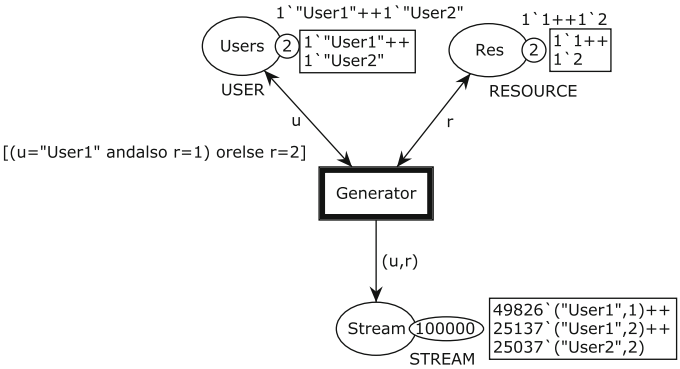


Fig. 6. Simulation results for the stream generator with guard

of valid user/resource pairs. However, we should also notice, that although the transition is fired 100,000 times, number of tokens in the output place is about 75,000. It is a result of the fact, that about 25,000 times when transition is fired for invalid user/resource pair, an empty token is produced. In this model the transition is fired for every possible mapping between users and resources, but invalid mappings do not produce results in the output place. This affects total number of resulting tokens and also distribution of the result values.

6 Explaining Behaviour of the Simulator

The results of simulations presented in the previous section can be explained by detailed description of solutions implemented in CPN simulators used in CPN Tools and its predecessor – Design/CPN [15]. The paper describes details of data structures and algorithms that ensure efficient simulation of CPN by Design/CPN software. In particular, it describes an approach to quickly find

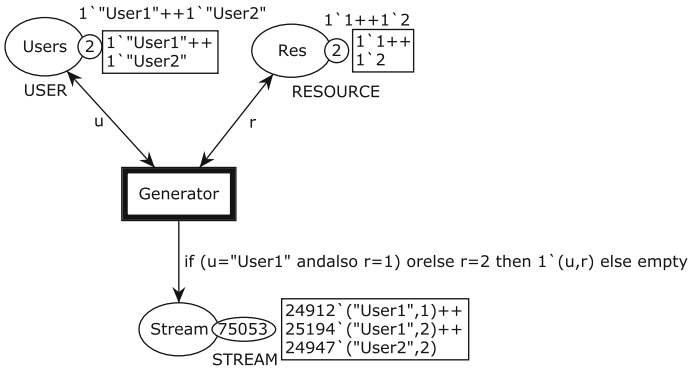


Fig. 7. Simulation results for the stream generator with restriction on the output arc

a variable binding that enables given CPN transition. One of the assumptions is that a transition should be fired immediately, whenever a valid binding is found. Thus the simulator does not generate all possible enabling bindings to select one of them randomly with uniform probability distribution. Instead it is focused on the most efficient method to find a variable binding that satisfies (arc- and guard-) expressions connected with the transition.

In the case of our model with guard expression (Fig. 3), the simulator first binds variable `r` to a randomly selected token from *Res* place, since no expression limits value of this variable. Thereafter, variable `u` is bound to a value that satisfies the guard expression. In the model from Fig. 4 any binding can be used to fire the *Generator* transition, but some bindings do not produce tokens in output place. Thus, having no hints as to which bindings are more desired, simulator randomly binds both variables to tokens from proper places, fires the transition and for about 25% of firings does not produce any output token.

In relation to probability of selecting specific binding, authors of the Design/CPN tool state that *we also need to ensure that the search algorithm is, in some sense, fair so that every enabled binding has a non-zero probability of being used* [15]. Thus the simulator by design does not guarantee an even distribution of selected bindings. It is not authors' concern to ensure that every enabled binding has equal probability of being selected, but to ensure, that every enabled binding can possibly be used at all. This is consistent with the notion of *fairness* used in concurrent processing. However, in particular applications of CPN simulator it can have important impact on obtained results when particular distribution of results is desired. Actually, the analysed case shows that the simulator performs better then just to ensure *non-zero probability* of using a binding. Depending on the way the model was implemented, the usage of tokens to fire the transition can be considered even. Although in some cases the evenness may produce counter-intuitive results and this fact should be taken into consideration.

7 Summary

Our paper discusses a particular behaviour of CPN Tools software toolkit during a sample data stream generation. Two models of the generator, which should behave similarly, produce different amounts of tokens. This may have serious consequences during the performance evaluation analysis of systems where such input data streams are applied. For example, the improper system assessment may occur, especially in the simulations where data streams are used for bottlenecks detection. The user of the model may be under the impression that he produces the assumed amount of the input data, whereas the system under the test acquires only a part of tokens stream.

We have shown that the behaviour of the TCPN models, although might be counter intuitive, is consistent with the definition of the TCPN. We discussed and explained obtained distribution of tokens in the light of this definition and considering assumptions of the Petri net simulator being part of TCPN Tools.

Described results should be taken into consideration while modelling real applications in order to improve reliability and thus practical usefulness of the analysis. They will improve e.g. future versions of the industrial system described in [10].

References

1. Bause, F.: Queueing Petri Nets - a formalism for the combined qualitative and quantitative analysis of systems. In: PNPM 1993, pp. 14–23. IEEE Press (1993)
2. Bolch, G., Greiner, S., de Meer, H., Trivedi, K.S.: Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications, 2nd edn. Wiley, Hoboken (2006)
3. Caprihan, R., Kumar, A., Stecke, K.E.: Evaluation of the impact of information delays on flexible manufacturing systems performance in dynamic scheduling environments. Int. J. Adv. Manuf. Technol. **67**, 311–338 (2013)
4. Chen, H., Yao, D.D.: Fundamentals of Queueing Networks. Springer, New York (2001). https://doi.org/10.1007/978-1-4757-5301-1
5. Dattatreya, G.R., Sahni, S.: Performance Analysis of Queuing and Computer Networks. Chapman and Hall, New York (2008)
6. Davidrajuh, R.: Distributed workflow based approach for eliminating redundancy in virtual enterprising. J. Supercomput. **63**(1), 107–125 (2013)
7. Diaz, M. (ed.): Petri Nets, Fundamental Models, Verification and Applications. Wiley, Hoboken (2009)
8. Fortier, J.P., Michel, H.E.: Computer Systems Performance Evaluation and Prediction. Elsevier, Burlington (2003)
9. Girault, C., Valk, R.: Petri Nets for Systems Engineering. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-662-05324-9
10. Jamro, M., Rzonca, D., Rząsa, W.: Testing communication tasks in distributed control systems with SysML and Timed Colored Petri Nets model. Comput. Ind. **71**, 77–87 (2015)
11. Jensen, K., Kristensen, L.M.: Coloured Petri Nets Modelling and Validation of Concurrent Systems. Springer, Heidelberg (2009). https://doi.org/10.1007/b95112

12. Kounev, S.: Performance modelling and evaluation of distributed component-based systems using Queuing Petri Nets. IEEE Trans. Softw. Eng. **32**(7), 486–502 (2006)
13. Kouvatsos, D. (ed.): Performance Evaluation of ATM Networks. Kluwer, Boston (2000)
14. Lenz, K., Mevius, M., Oberweis, A.: Process-oriented business performance management with Petri nets. In: The 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service, EEE 2005, Proceedings, pp. 89–92. IEEE (2005)
15. Mortensen, K.H.: Efficient data-structures and algorithms for a Coloured Petri Nets simulator. In: Proceedings of the 3rd Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools, Aarhus, Denmark, 29–31 August 2001
16. Noorshams, Q., Rostami, K., Kounev, S., Reussner, R.: Modeling of I/O performance interference in virtualized environments with Queueing Petri Nets. In: 22nd International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems. IEEE (2014)
17. Olejnik, R.: Modelling of half-duplex radio access for HopeMesh experimental WMN using Petri Nets. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2014. CCIS, vol. 431, pp. 108–117. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07941-7_11
18. Rak, T.: Performance modeling using Queueing Petri Nets. In: Gaj, P., Kwiecień, A., Sawicki, M. (eds.) CN 2017. CCIS, vol. 718, pp. 321–335. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59767-6_26
19. Rak, T., Samolej, S.: Simulation and performance analysis of distributed internet systems using TCPNs. Informatica Int. J. Comput. Inf. **33**(4), 405–415 (2009)
20. Rząsa, W.: Combining timed Colored Petri Nets and real TCP implementation to reliably simulate distributed applications. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2009. CCIS, vol. 39, pp. 79–86. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02671-3_9
21. Rząsa, W.: Timed Colored Petri Net based estimation of efficiency of the grid applications. Ph.D. thesis, AGH University of Science and Technology, Kraków, Poland (2011)
22. Rząsa, W.: Synchronization algorithm for timed Colored Petri Nets and Ns-2 simulators. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2013. CCIS, vol. 370, pp. 1–10. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38865-1_1
23. Rząsa, W.: Simulation-based analysis of a platform as a service infrastructure performance from a user perspective. In: Gaj, P., Kwiecień, A., Stera, P. (eds.) CN 2015. CCIS, vol. 522, pp. 182–192. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19419-6_17
24. Rząsa, W., Rzonca, D.: Event-driven approach to modeling and performance estimation of a distributed control system. In: Gaj, P., Kwiecień, A., Stera, P. (eds.) CN 2016. CCIS, vol. 608, pp. 168–179. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39207-3_15
25. Rząsa, W., Jamro, M., Rzonca, D.: Improving accuracy of a network model basing on the case study of a distributed system with a mobile application and an API. In: Gaj, P., Kwiecień, A., Sawicki, M. (eds.) CN 2017. CCIS, vol. 718, pp. 14–27. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59767-6_2
26. Rząsa, W.: Predicting performance in a PaaS environment: a case study for a web application. Comput. Sci. **18**(1), 21–39 (2017)

27. Rzońca, D., Trybus, B.: Hierarchical Petri Net for the CPDev virtual machine with communications. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2009. CCIS, vol. 39, pp. 264–271. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02671-3_31
28. Rzońca, D., Sadolewski, J., Trybus, B.: OPC data acquisition server for CPDev engineering environment. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2010. CCIS, vol. 79, pp. 315–321. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13861-4_33
29. Rzońca, D., Stec, A., Trybus, B.: Data acquisition server for mini distributed control system. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2011. CCIS, vol. 160, pp. 398–406. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21771-5_43
30. Samolej, S., Szmuc, T.: HTCPNs-based tool for web-server clusters development. In: Huzar, Z., Koci, R., Meyer, B., Walter, B., Zendulka, J. (eds.) CEE-SET 2008. LNCS, vol. 4980, pp. 131–142. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22386-0_10
31. Samolej, S., Szmuc, T.: HTCPNs-based modelling and evaluation of dynamic computer cluster reconfiguration. In: Szmuc, T., Szpyrka, M., Zendulka, J. (eds.) CEE-SET 2009. LNCS, vol. 7054, pp. 97–108. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28038-2_8
32. Samolej, S., Szmuc, T.: Web-server systems HTCPNs-based development tool application in load balance modelling, e-Informatica. Softw. Eng. J. **3**(1), 139–153 (2009)
33. Samolej, S., Szmuc, T.: HTCPNs-based analysis of priority-based internet requests scheduling. Przeglad Elektrotechniczny **86**(2010), 174–178 (2010)
34. Wells, L., et al.: Simulation based performance analysis of web servers. In: Proceedings of the 9th International Workshop on Petri Nets and Performance Models, p. 59. IEEE (2001)
35. Wells, L.: Performance analysis using CPN tools. In: Proceedings of the 1st International Conference on Performance Evaluation Methodologies and Tools, p. 59 (2006)

Transient Queueing Delay in a Finite-Buffer Batch-Arrival Model with Constant Repeated Vacations

Wojciech M. Kempa(✉) and Rafał Marjasz

Faculty of Applied Mathematics, Silesian University of Technology,
23 Kaszubska Street, 44-100 Gliwice, Poland
{wojciech.kempa,rafal.marjasz}@polsl.pl

Abstract. A finite-buffer single-channel queueing model with batch Poisson arrivals and generally-distributed processing times is considered, in which a variant of a vacation policy is implemented. Namely, every time when the service station becomes idle, a number of constant vacation times are being initialized repeatedly, until at least one packet will income into the accumulating buffer. During the whole vacation period the processing of packets is suspended. Applying the analytical method based on the idea of embedded Markov chain and linear-algebraic approach, a compact-form representation for the Laplace transform of the queueing delay tail distribution is found. The considered queueing system can be utilized in performance evaluation of wireless network nodes with energy saving mechanism based on constant repeated vacations.

Keywords: Energy saving · Multiple vacation policy
Queueing delay · Queueing system · Transient distribution

1 Preliminaries

Queueing systems with finite buffer capacities are intensively studied nowadays due to their wide applications, in particular in modelling the operation of computer and telecommunication networks' nodes. Models with different-types limitations in the access to the service station seem to be of particular importance, due to their potential using in energy saving modelling, being one of the most essential problems in wireless communication. In this case each busy period of the considered queueing system can be treated as an active period in the operation of the wireless network node (e.g. wireless sensor network) while, similarly, each idle time may be considered as a sleep (power saving) period.

In the literature different-type policies are proposed for supporting energy saving modelling. One of them is the so called multiple vacation policy (MVP in short). The sense of MVP is in that the service station (e.g. the radio transmitter/receiver of the wireless sensor network's node) takes a number of repeated independent vacation periods, every time when the queue of buffered packets

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 311–320, 2018.
https://doi.org/10.1007/978-3-319-92459-5_25

being directed to the node becomes empty. Successive vacation periods are being initialized until, at the end of one of them, at least one packet will be detected (so at least one packet will income into the accumulating buffer). For example, the mechanism of repeated vacations implemented into the $M/G/1$-type system is proposed in [14] (see also [1,18]) in modelling of type I energy-saving mode in IEEE 802.16e standard, and some performance measures are found there.

It is easy to note that theoretical results obtained for different-type queueing models with some service limitations are given in the stationary regime, i.e. they relate to the stochastic characteristics in the case of $t \to \infty$. In practice, there are some situations in which transient analysis seems to be more recommended. For example, in the case of rather low intensity of input traffic (as it can be observed e.g. in some wireless sensor networks), in the case of performance evaluation of the system shortly after its starting (or after the application of a new control mechanism) or after the breakdown that destabilizes the system's operation.

In the paper we study the transient queueing delay in the $M^X/G/1/N$-type finite-buffer queueing model with batch arrivals and the power saving mechanism based on a variant of MVP in that successive server vacations in a single MVP period have constant lengths. Queueing delay (called also virtual waiting time) at arbitrary fixed time epoch t takes a random value equal to the waiting time of the packet occurring in the system exactly at time t. Since this time need not be a "real" arrival moment, hence the term "virtual". Applying the theoretical approach based on the idea of embedded Markov chain, the formula of total probability and linear algebra, the representation for the Laplace transform of the tail cumulative distribution function (CDF for short) of the queueing delay at given time t is found, conditioned by the initial level of buffer saturation.

The review of steady-state results for different vacation queueing models can be found in [4,19]. An infinite-buffer queueing model with $BMAP$-type input stream and server vacations is investigated in [17]. Analytical results for finite-capacity systems with server vacations can be found e.g. in [5,6,15,16,20]. In particular, one can find the formulae for the stationary waiting-time distribution in the case of autocorrelated arrival stream in [15,16]. The formula for the queueing delay distribution in the system with MVP and Poisson arrivals in equilibrium is derived in [20]. More complex vacation policies can be found in [2,21]. In [2] $M/G/1$-type system with server activity controlled by a timer is studied in the stationary case. The model reduces to the "usual" MVP queue with a zero-length timer duration. A queueing system with batch Poisson arrivals and server vacations controlled by the Randomly Timed Gated protocol is investigated in [21].

Time-dependent (transient) results on the queueing delay distribution in general-type models with batch arrivals and infinite buffers can be found e.g. in [7,8]. The approach used there is based on integral equations and Wiener-Hopf factorization technique. Finite-capacity queue with MVP is studied in [9]. Analytical results on non-stationary queueing delay in systems with more complex control mechanisms can be found in [10–12].

The remaining part of the article is organized as follows. In the next section we give the detailed description of the considered queueing model. In Sect. 3 we obtain a system of integral equations for conditional queueing delay tail CDF. The corresponding system for Laplace transforms is found in Sect. 4. The solution of the system, written in a compact form (main result) is given in Sect. 5.

2 Model Description

An $M^X/G/1/N$-type queueing model is considered in which packets arrive according to a compound Poisson process with rate λ and generally-distributed group sizes. Namely, exactly k packets arrive simultaneously with probability p_k, where $\sum_{k=1}^{\infty} p_k = 1$. It is assumed that packets are being processed one by one, under the FIFO service discipline, with a general-type cumulative distribution function (CDF for short) $F(\cdot)$ of processing time. The accumulating buffer has capacity of $N-1$ packets, so the maximal system state is assumed to be N. Every time when the service station becomes idle, a variant of a multiple vacation policy is being initialized: successive constant vacations of length $D > 0$ are being started repeatedly until at least one waiting packet will be detected in the accumulating buffer at the end of one of them. In such a case, at the same moment the processing restarts and so on.

3 Transient Equations for Queueing Delay Conditional Distribution

Let us denote by $v(t)$ queueing delay at time t, namely the waiting time of a packet entering (hypothetically) exactly at time t.

Introduce the queueing delay conditional tail CDF as follows:

$$V_n(t,x) \stackrel{def}{=} \mathbf{P}\{v(t) > x \,|\, X(0) = n\}, \quad t > 0,\, x > 0, \tag{1}$$

where $X(0)$ stands for the system state at the starting epoch $t = 0$.

Consider, firstly, the case of the system being empty before its opening and fix a moment $t > 0$. Three possible and mutually excluding situation may then occur:

- the first multiple vacation period ends before t (A);
- the first multiple vacation period ends after t but the system is not empty at t (B);
- the first packet enters after time t (C).

Let us note that the following equation is then satisfied:

$$\mathbf{P}\{\big(v(t) > x\big) \cap A \,|\, X(0) = 0\} = \sum_{k=0}^{\infty} \int_0^t \lambda e^{-\lambda y} I\{kD \le y < (k+1)D < t\}$$

$$\times \left[\sum_{i=1}^{N-1} p_i \left(\sum_{r=0}^{N-i-1} \sum_{j=0}^{r} \frac{\{\lambda[(k+1)D-y]\}^j}{j!} e^{-\lambda[(k+1)D-y]} p_r^{j*} V_{i+r}(t-(k+1)D,x) \right.\right.$$
$$\left. + V_N(t-(k+1)D,x) \sum_{r=N-i}^{\infty} \sum_{j=0}^{r} \frac{\{\lambda[(k+1)D-y]\}^j}{j!} e^{-\lambda[(k+1)D-y]} p_r^{j*} \right)$$
$$\left. + V_N(t-(k+1)D,x) \sum_{i=N}^{\infty} p_i \right] dy, \tag{2}$$

where p_r^{j*} denotes the rth term of the j-fold convolution of the sequence (p_k) with itself and $I\{\Pi\}$ is the indicator of the random event Π.

Similarly, we have

$$\mathbf{P}\{\big(v(t) > x\big) \cap B \,|\, X(0) = 0\} = \sum_{k=0}^{\infty} \int_0^t \lambda e^{-\lambda y} I\{kD \le y,\, (k+1)D > t\}$$
$$\times \sum_{i=1}^{N-1} p_i \sum_{r=0}^{N-i-1} \sum_{j=0}^{r} \frac{[\lambda(t-y)]^j}{j!} e^{-\lambda(t-y)} p_r^{j*} \overline{F}^{(i+r)*}(x-(k+1)D+t)dy, \tag{3}$$

where $\overline{F}^{j*}(u) \stackrel{def}{=} 1 - F^{j*}(u)$ and $F^{j*}(u)$ stands for the i-fold Stieltjes convolution of the CDF $F(\cdot)$ with itself and is defined as

$$F^{0*}(t) = 1, \;\; F^{1*}(t) = F(t), \;\; F^{(n+1)*}(t) = \int_0^t F^{n*}(t-y)dF(y), \tag{4}$$

where $t > 0$ and $n \ge 1$.

Evidently, $\mathbf{P}\{\big(v(t) > x\big) \cap C \,|\, X(0) = 0\} = 0$, since then the packet entering at time t has no waiting time.

Due to the fact that

$$V_0(t,x) = \mathbf{P}\{\big(v(t) > x\big) \cap A \,|\, X(0) = 0\}$$
$$+ \mathbf{P}\{\big(v(t) > x\big) \cap B \,|\, X(0) = 0\} + \mathbf{P}\{\big(v(t) > x\big) \cap C \,|\, X(0) = 0\}, \tag{5}$$

evaluating integrals in (2)–(3), we obtain

$$V_0(t,x) = \sum_{k=0}^{\infty} e^{-\lambda(k+1)D} I\{(k+1)D < t\}$$
$$\times \left[\sum_{i=1}^{N-1} p_i \left(\sum_{r=0}^{N-i-1} \sum_{j=0}^{r} \frac{(\lambda D)^{j+1}}{(j+1)!} p_r^{j*} V_{i+r}(t-(k+1)D,x) \right.\right.$$
$$\left.\left. + V_N(t-(k+1)D,x) \sum_{r=N-i}^{\infty} \sum_{j=0}^{r} \frac{(\lambda D)^{j+1}}{(j+1)!} p_r^{j*} \right) + V_N(t-(k+1)D,x) \sum_{i=N}^{\infty} p_i \right]$$

$$+ e^{-\lambda t} \sum_{k=0}^{\infty} I\{(k+1)D > t\} \sum_{i=1}^{N-1} p_i \sum_{r=0}^{N-i-1} \overline{F}^{(i+r)*}(x-(k+1)D+t)$$
$$\times \sum_{j=0}^{r} \frac{[\lambda(t-kD)]^{j+1}}{(j+1)!} p_r^{j*}. \tag{6}$$

Assume now that the buffer is not empty at the starting time. In such a case successive departure epochs are Markovian moments in the evolution of the system (see e.g. [3]). Applying the law of total probability with respect to the first departure epoch $y > 0$ after the opening of the system, we obtain

$$V_n(t,x) = \int_0^t \left[\sum_{i=0}^{N-n-1} \sum_{j=0}^{i} \frac{(\lambda y)^j}{j!} e^{-\lambda y} p_i^{j*} V_{n+i-1}(t-y,x) \right.$$
$$\left. + V_{N-1}(t-y,x) \sum_{i=N-n}^{\infty} \sum_{j=0}^{i} \frac{(\lambda y)^j}{j!} e^{-\lambda y} p_i^{j*} \right] dF(y) + \int_t^{\infty} \overline{F}^{(n-1)*}(x-y+t) dF(y), \tag{7}$$

where $1 \le n \le N$.

4 Linear Equations for Transforms

In this section we transform the original system of Eqs. (6)–(7) to the corresponding linear one written for Laplace transforms.

Indeed, let us introduce the following nomenclature:

$$v_n(s,x) \stackrel{def}{=} \int_0^{\infty} e^{-st} V_n(t,x) dt; \tag{8}$$

$$a_r(s) \stackrel{def}{=} \int_0^{\infty} e^{-(\lambda+s)t} \sum_{j=0}^{r} \frac{(\lambda t)^j}{j!} p_r^{j*} dF(t); \tag{9}$$

$$\alpha_r(s) \stackrel{def}{=} \frac{e^{-(\lambda+s)D}}{1-e^{-(\lambda+s)D}} \sum_{j=0}^{r} \frac{(\lambda D)^{j+1}}{(j+1)!} p_r^{j*}; \tag{10}$$

$$b_n(s,x) \stackrel{def}{=} \int_{t=0}^{\infty} e^{-st} dt \int_{y=t}^{\infty} \overline{F}^{(n-1)*}(x-y+t) dF(y), \tag{11}$$

$$\beta(s) \stackrel{def}{=} \frac{e^{-(\lambda+s)D}}{1-e^{-(\lambda+s)D}} \left[\sum_{i=1}^{N-1} p_i \sum_{r=N-i}^{\infty} \sum_{j=0}^{r} p_r^{j*} \frac{(\lambda D)^{j+1}}{(j+1)!} + \sum_{i=N}^{\infty} p_i \right] \tag{12}$$

and

$$\gamma(s,x) \stackrel{def}{=} \int_0^{\infty} e^{-(\lambda+s)t} \sum_{k=0}^{\infty} I\{(k+1)D > t\}$$
$$\sum_{i=1}^{N-1} p_i \sum_{r=0}^{N-i-1} \overline{F}^{(i+r)*}(x-(k+1)D+t) \sum_{j=0}^{r} \frac{[\lambda(t-kD)]^{j+1}}{(j+1)!} p_r^{j*} dt. \tag{13}$$

Now, the Eqs. (6)–(7) can be rewritten as follows:

$$v_0(s,x) = \sum_{i=1}^{N-1} p_i \sum_{r=0}^{N-i-1} \alpha_r(s) v_{i+r}(s,x) + v_N(s,x)\beta(s) + \gamma(s,x), \tag{14}$$

$$v_n(s,x) = \sum_{i=0}^{N-n-1} a_i(s) v_{n+i-1}(s,x) + v_{N-1}(s,x) \sum_{i=N-n}^{\infty} a_i(s) + b_n(s,x), \tag{15}$$

where $1 \leq n \leq N$.

Let us introduce into Eqs. (14)–(15) the following substitution:

$$w_n(s,x) \stackrel{def}{=} v_{N-n}(s,x), \tag{16}$$

where $0 \leq n \leq N$.

After this transformation the system (14)–(15) takes the following shape:

$$\sum_{k=-1}^{n} a_{k+1}(s) w_{n-k}(s,x) - w_n(s,x) = \phi_n(s,x), \tag{17}$$

where $0 \leq n \leq N-1$, and

$$w_N(s,x) = \sum_{i=1}^{N-1} p_i \sum_{r=0}^{N-i-1} \alpha_{N-i-r}(s) w_r(s,x) + w_0(s,x)\beta(s) + \gamma(s,x), \tag{18}$$

where the functional sequence $\big(\phi_n(\cdot,\cdot)\big)$ is defined in the following way:

$$\phi_n(s,x) \stackrel{def}{=} a_{n+1}(s) w_0(s,x) - w_1(s,x) \sum_{k=n+1}^{\infty} a_k(s) - b_{N-n}(s,x). \tag{19}$$

5 Compact-Form Solution

The general solution of the algebraic system of the form (17) but with infinite number of equations (written for $n \geq 0$) can be written as follows (see [13]):

$$w_n(s,x) = C(s,x) R_{n+1}(s) + \sum_{k=0}^{n} R_{n-k}(s) \phi_k(s,x), \tag{20}$$

where $n \geq 0$ and the sequence $\big(R_n(s)\big)$ is defined by coefficients $a_k(s)$ as follows. Introduce the following generating functions:

$$R(s,z) \stackrel{def}{=} \sum_{k=0}^{\infty} z^k R_k(s), \quad A(s,z) \stackrel{def}{=} \sum_{k=0}^{\infty} z^k a_k(s), \quad |z| < 1. \tag{21}$$

Now the following relationship is true:

$$R(s,z) = \frac{z}{A(s,z) - z}. \tag{22}$$

To find successive terms of the sequence $\big(R_k(s)\big)$ we can use the Maclaurin expansion. Indeed, if we denote $Q(s,z) \stackrel{def}{=} \frac{z}{A(s,z)-z}$, then we obtain

$$\sum_{k=0}^{\infty} z^k R_k(s) = \sum_{k=0}^{\infty} z^k \frac{\partial^k Q}{\partial z^k}\Big|_{(s,0)} \frac{1}{k!} \tag{23}$$

and hence

$$R_k(s) = \frac{\partial^k Q}{\partial z^k}\Big|_{(s,0)} \frac{1}{k!}. \tag{24}$$

In particular, we have $R_1(s) = \frac{1}{a_0(s)}$.

Since the number of equations in (17) is finite, we can use the Eq. (18) as a specific-type boundary condition to find $C(s,x)$ explicitly. In fact, instead of $C(s,x)$, we will find the formula for $w_0(s,x)$. Indeed, the relationship between $C(s,x)$ and $w_0(s,x)$ can be easily obtained from (20). Substituting $n=0$, we get

$$C(s,x) = w_0(s,x)[R_1(s)]^{-1} = w_0(s,x)a_0(s). \tag{25}$$

Similarly, since $\sum_{k=0}^{\infty} a_k(s) = f(s)$, where $f(\cdot)$ is the Laplace transform of CDF $F(\cdot)$, we obtain from (19), written at $n=0$,

$$\phi_0(s,x) = a_1(s)w_0(s,x) - w_1(s,x)[f(s) - a_0(s)] - b_N(s,x). \tag{26}$$

From (17), taking $n=0$, we obtain

$$\phi_0(s,x) = a_0(s)w_1(s,x) + a_1(s)w_0(s,x) - w_0(s,x). \tag{27}$$

Comparing the right sides of (26) and (27), we get $w_1(s,x)$ in a function of $w_0(s,x)$, namely

$$w_1(s,x) = [f(s)]^{-1}[w_0(s,x) - b_N(s,x)]. \tag{28}$$

The remaining task is to find $w_0(s,x)$ explicitly. In order to do it, we write the representation for $w_N(s,x)$ in two different forms: in the first one we use the general form of solution (20) and in the second one we use the Eq. (18). From (20) written at $n=N$ (having in mind (25) and (28)) we get

$$w_N(s,x) = w_0(s,x)a_0(s)R_{N+1}(s) + \sum_{k=0}^{N} R_{N-k}(s)$$
$$\times \Big\{a_{k+1}(s)w_0(s,x) - [f(s)]^{-1}[w_0(s,x) - b_N(s,x)] \sum_{i=k+1}^{\infty} a_i(s) - b_{N-k}(s,x)\Big\}. \tag{29}$$

Similarly, applying (25) in (18), we get

$$w_N(s,x) = \sum_{i=1}^{N-1} p_i \sum_{r=0}^{N-i-1} \alpha_{N-i-r}\Big\{w_0(s,x)a_0(s)R_{r+1}(s) + \sum_{k=0}^{r} R_{r-k}(s)\Big[a_{k+1}(s)w_0(s,x) - [f(s)]^{-1}[w_0(s,x) - b_N(s,x)] \sum_{i=k+1}^{\infty} a_i(s) - b_{N-k}(s,x)\Big]\Big\} + w_0(s,x)\beta(s) + \gamma(s,x). \tag{30}$$

Comparing the right sides of (29) and (30), we eliminate $w_0(s,x)$ as follows:

$$w_0(s,x) = T_1(s,x)T_2(s,x), \tag{31}$$

where

$$T_1(s,x) \stackrel{def}{=} \Bigg\{ a_0(s)R_{N+1}(s) + \sum_{k=0}^{N} R_{N-k}(s)\Big[a_{k+1}(s) - \big(f(s)\big)^{-1} \sum_{i=k+1}^{\infty} a_i(s)\Big] - \sum_{i=1}^{N-1} p_i \sum_{r=0}^{N-i-1} \alpha_{N-i-r}(s)\Bigg[a_0(s)R_{r+1}(s) + \sum_{k=0}^{r} R_{r-k}(s)\Big[a_{k+1}(s) - \big(f(s)\big)^{-1} \sum_{i=k+1}^{\infty} a_i(s)\Big]\Bigg] - \beta(s)\Bigg\}^{-1} \tag{32}$$

and

$$T_2(s,x) \stackrel{def}{=} \sum_{i=1}^{N-1} p_i \sum_{r=0}^{N-i-1} \alpha_{N-i-r}(s) \sum_{k=0}^{r} R_{r-k}(s) \times \Big\{ \big[f(s)\big]^{-1} b_N(s,x) \sum_{i=k+1}^{\infty} a_i(s) - b_{N-k}(s,x)\Big\} + \gamma(s,x) - \sum_{k=0}^{N} R_{N-k}(s)\Big\{ \big[f(s)\big]^{-1} b_N(s,x) \sum_{i=k+1}^{\infty} a_i(s) - b_{N-k}(s,x)\Big\}. \tag{33}$$

Now, collecting the formulae (16), (20), (25) and (31), we obtain the following main result:

Theorem 1. *The Laplace transform of the queueing delay conditional tail CDF in the considered queueing system with constant repeated vacations can be written in the following way:*

$$v_n(s,x) = \Big\{a_0(s)R_{N-n+1}(s) + \sum_{k=0}^{N-n} R_{N-n-k}(s)\Big[a_{k+1}(s) - \big[f(s)\big]^{-1} \sum_{i=k+1}^{\infty} a_i(s)\Big]\Big\} T_1(s,x)T_2(s,x)$$

$$+\sum_{k=0}^{N-n} R_{N-n-k}(s)\Big\{\big[f(s)\big]^{-1} b_N(s,x)\sum_{i=k+1}^{\infty} a_i(s) - b_{N-k}(s,x)\Big\}, \qquad (34)$$

where $0 \le n \le N$ *and the formulae for* $a_k(s)$, $b_k(s,x)$, $R_k(s)$, $T_1(s,x)$ *and* $T_2(s,x)$ *are given in (9), (11), (24), (32) and (33), respectively.*

References

1. Alouf, S., Altman, E., Azad, A.: M/G/1 queue with repeated inhomogeneous vacations applied to IEEE 802.16e power saving. In: Proceedings of ACM SIGMETRICS 2008, Performance Evaluation Review, vol. 36, pp. 451–452 (2008)
2. Boxma, O.J., Schlegel, S., Yechiali, Y.: A note on an M/G/1 queue with a waiting server, timer and vacations. Am. Math. Soc. Transl. Ser. **2**(207), 25–35 (2002)
3. Cohen, J.W.: The Single Server Queue. North-Holland Publishing Company, Amsterdam-New York-Oxford (1982)
4. Doshi, B.T.: Queueing systems with vacations - a survey. Queueing Syst. **1**, 29–66 (1986)
5. Gupta, U.C., Banik, A.D., Pathak, S.S.: Complete analysis of $MAP/G/1/N$ queue with single (multiple) vacation(s) under limited service discipline. J. Appl. Math. Stoch. Anal. **3**, 353–373 (2005)
6. Gupta, U.C., Sikdar, K.: Computing queue length distributions in $MAP/G/1/N$ queue under single and multiple vacation. Appl. Math. Comput. **174**(2), 1498–1525 (2006)
7. Kempa, W.M.: The virtual waiting time for the batch arrival queueing systems. Stoch. Anal. Appl. **22**(5), 1235–1255 (2004)
8. Kempa, W.M.: Some results for the actual waiting time in batch arrival queueing systems. Stoch. Models **26**(3), 335–356 (2010)
9. Kempa, W.M.: Transient workload distribution in the M/G/1 finite-buffer queue with single and multiple vacations. Ann. Oper. Res. **239**(2), 381–400 (2016)
10. Kempa, W.M., Kurzyk, D.: Analysis of transient virtual delay in a finite-buffer queueing model with generally distributed setup times. In: Czachórski, T., Gelenbe, E., Grochla, K., Lent, R. (eds.) ISCIS 2016. CCIS, vol. 659, pp. 175–184. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-47217-1_19
11. Kempa, W.M., Paprocka, I., Kalinowski, K., Grabowik, C., Krenczyk, D.: Study on transient queueing delay in a single-channel queueing model with setup and closedown times. In: Dregvaite, G., Damasevicius, R. (eds.) ICIST 2016. CCIS, vol. 639, pp. 464–475. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46254-7_37
12. Kempa, W.M.: Queueing delay in a finite-buffer model with failures and bernoulli feedback. In: Świątek, J., Borzemski, L., Wilimowska, Z. (eds.) ISAT 2017. AISC, vol. 656, pp. 229–238. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-67229-8_21
13. Korolyuk, V.S.: Boundary-Value Problems for Complicated Poisson Processes. Naukova Dumka, Kiev (1975)
14. Mancuso, V., Alouf, S.: Analysis of power saving with continuous connectivity. Comput. Netw. **10**, 2481–2493 (2012)
15. Niu, Z., Takahashi, Y.: A finite-capacity queue with exhaustive vacation/closedown/setup times and Markovian arrival processes. Queueing Syst. **31**, 1–23 (1999)

16. Niu, Z., Shu, T., Takahashi, Y.: A vacation queue with setup and close-down times and batch Markovian arrival processes. Perform. Eval. **54**(3), 225–248 (2003)
17. Saffer, Z., Telek, M.: Analysis of $BMAP/G/1$ vacation model of non-$M/G/1$-type. In: Thomas, N., Juiz, C. (eds.) EPEW 2008. LNCS, vol. 5261, pp. 212–226. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87412-6_16
18. Seo, J., Lee, S., Park, N., Lee, H., Cho C.: Performance analysis of sleep mode operation in IEEE 802.16e. In: Proceedings of the 60th Vehicular Technology Conference, VTC2004-Fall (Los Angeles), vol. 2, pp. 1169–1173 (2004)
19. Takagi, H.: Queueing Analysis, vol. 1: Vacation and Priority Systems, vol. 2. Finite Systems. North-Holland, Amsterdam (1993)
20. Takagi, H.: M/G/1/N queues with server vacations and exhaustive service. Oper. Res. **42**(5), 926–939 (1994)
21. Yechiali, Y., Shomrony, M.: Burst arrival queues with server vacations and random timers. Math. Methods Oper. Res. **53**(1), 117–146 (2001)

Cybersecurity and Quality of Service

Cache Enhanced Anonymity Systems Against Probabilistic Attacks

Huabo Lu and Rajiv Bagai(✉)

Department of Electrical Engineering and Computer Science,
Wichita State University, Wichita, KS 67260-0083, USA
{hxlu,rajiv.bagai}@wichita.edu

Abstract. The technique of data caching is being increasingly recognized for its ability to increase the amount of anonymity provided by an anonymous communication network, and is expected to be widely adopted in the next generation of these systems. We present a method to measure the degree of anonymity remaining in a cache enhanced anonymity system after a probabilistic attack has been carried out on it. Our method determines the probability distribution induced by the attack on all possible communication patterns of being the true one, from which a system-wide anonymity metric is developed. The scope of our metric, in terms of the attacks it is applicable to, is far wider than that of existing methods.

Keywords: Online anonymity · Data caching · Probabilistic attacks
Measuring anonymity · Shannon entropy · Combinatorial matrix theory

1 Introduction

Being able to stay anonymous while performing online activities is highly preferred in many sensitive scenarios, like online voting, job or partner searching, evaluation of superiors at workplace, law enforcement undercover operations, etc. Anonymous communication networks, or *anonymity systems*, are computer networks that aim to provide anonymity to users engaged in such tasks. A popular architecture for such a system was initially proposed by Chaum [1], and a variety of different anonymous systems have been built since then, such as TOR [2] and Riffle [3].

The basic building block for anonymity systems is a *mix-node*, which takes in multiple messages from users, "mixes" those messages, and then sends them out in a way that conceals the identity of each output message's sender. The mixing action breaks easy linkage between incoming messages and corresponding outgoing messages. For example, shuffling the order of outgoing messages breaks message linkage by arrival/departure order, and encryption breaks message linkage on size or bit pattern since the same message after encryption possesses different such characteristics. Typically, an anonymity system has multiple mix-nodes, forming a mix-network. Users gain anonymity when their network traffic is routed by the system via carefully chosen mix-nodes.

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 323–332, 2018.
https://doi.org/10.1007/978-3-319-92459-5_26

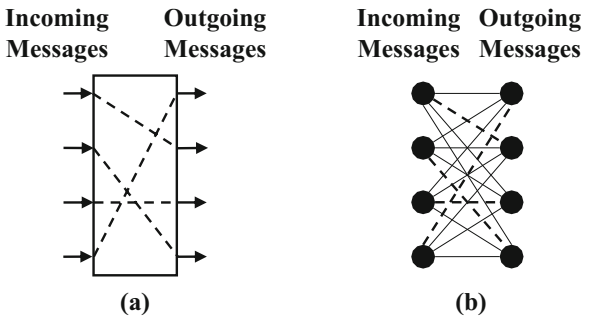


Fig. 1. (a) Block diagram of an anonymity system; (b) The bipartite graph of all possible communication patterns, hiding the true communication pattern.

Figure 1(a) shows a block diagram of an anonymity system that receives messages from its users and forwards them to their respective destinations. The dashed lines inside the system represent one-to-one linkages between its incoming and outgoing messages. These linkages are created internally by the system, with an intention to hide them from any external observer. We refer to these linkages as the system's *true communication pattern*. To an external observer, all linkages are possible, depicted by the complete bipartite graph of Fig. 1(b). The system's true communication pattern, although highlighted in this figure, is in fact hidden among all possible communication patterns, thereby providing anonymity to the users.

An adversary's goal is to break anonymity and determine, in whole or in part, the system's true communication pattern. Adversaries are usually considered to be *global* and *passive*, as explained in Diaz et al. [4], Edman et al. [5] and Bagai et al. [6]. The global attribute means adversaries are able to watch all incoming and outgoing traffic on the edges of the system, while the internal mixing action is still hidden. The passive attribute rules out any potential manipulation or modifying actions on the network traffic, i.e. no adversaries control any mix-nodes or affect network traffic by any means.

There are several classes of global passive attacks on anonymity systems and we would like to mention two classes that are closely related to our work. One class contains *infeasibility* attacks, and the other contains *probabilistic* attacks. An infeasibility attack deduces the infeasibility of linkages between some input and output messages, ruling out the impossible ones, thereby reducing the number of possible communication patterns. A probabilistic attack assigns probabilities on each linkage, inducing a probability distribution on all possible patterns, thereby making the most likely ones stand out. Some examples of both kinds of attacks can be found in Edman et al. [5] and Bagai et al. [7]. Also shown in these is the fact that the class of infeasibility attacks is a very small subclass of the class of probabilistic attacks.

To mitigate loss of anonymity caused by attacks, especially in bidirectional communication, such as anonymous web browsing, data *caching* within the sys-

tem's mix-nodes has been proposed as an effective countermeasure, as by Shubina et al. [8] and Kim et al. [9]. This feature deploys caches within an anonymity system, and fills them with frequently requested contents. When one of the cached contents can satisfy an incoming message, the anonymity system needs no longer send out that incoming message, and serves it from its stored cached content. This feature reduces the number of outgoing messages, and satisfied incoming messages enjoy full anonymity since they are never sent out of the anonymity system. As a result, anonymity systems using message caching are more robust to attacks.

Bagai et al. [6] showed the effectiveness of caching against attacks, and developed a method to measure anonymity provided by a system under infeasibility attacks. However, they left development of such a method for probabilistic attacks as future work. In this paper, we undertake that task and propose a metric that measures anonymity provided by a cache-enhanced anonymity system in the aftermath of a probabilistic attack.

The rest of this paper is organized as follows. Section 2 lays down the model of the anonymity system and attacks we consider. Section 3 introduces the mathematical concepts of threads and diagonals upon which our metric is based. Our metric is developed in Sect. 4, and is compared with existing anonymity metrics in Sect. 5. Finally, Sect. 6 concludes our work and gives some directions for future work.

2 The System Model

Let $X = \{x_1, x_2, \ldots, x_m\}$ be the set of m input messages observed by an attacker having entered an anonymity system. Of these, some messages are served by the system from its cache, and only n messages, where $m \geq n$, are forwarded by the system to their respective destinations. Let $Y = \{y_1, y_2, \ldots, y_n\}$ be the set of output messages observed by the attacker having exited from that system. The bipartite graph of all possible communication patterns is the complete bipartite graph $K_{m,n}$ between the system's input and output messages.

As each message in Y must have originated as some message in X, the system's true communication pattern can be modeled as a function $f : Y \to X$, which is complete, one-to-one, but not necessarily onto, because $m \geq n$. This function is contained and hidden in the graph $K_{m,n}$ and the attacker's goal is to uncover it as much as possible from all such functions contained in that graph. To this end, the attacker uses whatever information is available about the system and/or messages to arrive at a probability distribution over all functions contained in the graph.

As an example of an attack, consider the simple anonymity system shown in Fig. 2(a), with two mix-nodes, M_1 and M_2, four input messages, and three output messages. The message from node M_1 to M_2 is internal to the network, and M_2 serves one of the three messages it receives from its internal cache. The complete bipartite graph of this system is shown in Fig. 2(b). Suppose each mix-node first serves any incoming message from its cache, if possible, then randomly

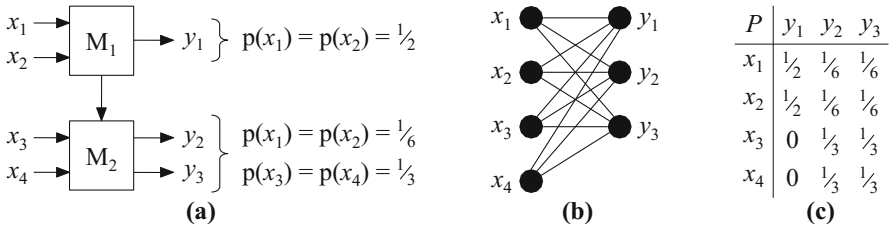


Fig. 2. (a) An example message flow via an anonymity system, observed by attacker; (b) System's complete graph, $K_{4,3}$; (c) Probability matrix resulting from this attack.

shuffles all other messages before sending them out. Thus, a message entering any mix-node is equally likely to appear as any of that node's output messages. If this behavior of mix-nodes is known to the attacker, and the entire message flow of the network (including internal messages) is also visible, the attacker can rationally arrive at probabilities on each edge of the complete graph, also shown next to the output messages in Fig. 2(a). These probability values can be represented in a 4×3 probability matrix P, as shown in Fig. 2(c). Any entry P_{ij} of this matrix is the probability of the system's input message x_i appearing as its output message y_j. Note that the sum of values in any row of P is *at most* 1, and the sum of values in any column of P is *exactly* 1. This is so because while some input messages may not appear at the output, each output message must have originated as some input message.

There are $\binom{4}{3} 3! = 24$ one-to-one functions from Y to X contained in $K_{4,3}$. While the system strives for each of these functions to appear to any observer to be equally possible as its true communication pattern, a probability matrix resulting from an attack induces a probability distribution over these functions. The stronger the attack (resulting from more observation of the system), the more non-uniform is this distribution, thereby exposing to a greater extent the true communication pattern. In the following sections, we first derive this distribution from a given probability matrix, and then develop a method for *measuring* the amount of system-wide anonymity remaining after the attack.

3 Threads and Diagonals

Recall that X is the set of m input messages, and Y the set of n output messages of the system. Given any $m \times n$ probability matrix P, we define a *thread* of P to be any collection of its entries that contains exactly one entry from each column of P. Each thread therefore has exactly n entries. Additionally, a thread of P is a *diagonal* if no two of its entries lie in the same row of P. Let $\mathcal{T}(P)$ and $\mathcal{D}(P)$ denote, respectively, the sets of all threads and diagonals of P. Note that, a thread corresponds to a function from Y to X, and a diagonal corresponds to a *one-to-one* function from Y to X. Clearly, P has m^n threads, of which $\binom{m}{n} n!$ are diagonals.

Let the *weight* of any thread t of P, denoted $\mathcal{W}(t)$, be the product of values in all entries of t. The sum of all thread weights is 1, as shown below.

Proposition 1. *For any probability matrix P,*

$$\sum_{t \in \mathcal{T}(P)} \mathcal{W}(t) = 1.$$

Proof. Let P be $m \times n$. By definitions and algebraic rearrangement we have,

$$\sum_{t \in \mathcal{T}(P)} \mathcal{W}(t) = \sum_{i_1=1}^{m} \sum_{i_2=1}^{m} \cdots \sum_{i_m=1}^{m} P_{i_1 1} P_{i_2 2} \cdots P_{i_m n} = \prod_{j=1}^{n} (P_{1j} + P_{2j} + \cdots + P_{mj}) = 1.$$

The last equality follows from the fact that the sum of each column of P is 1. □

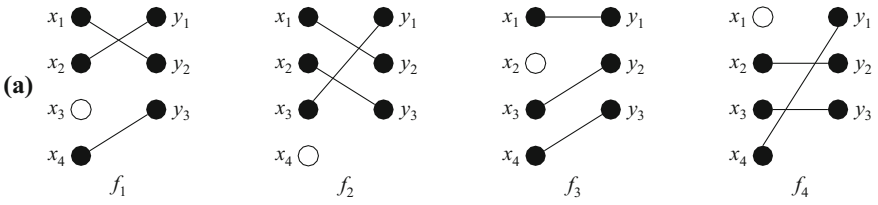
The sum of all diagonal weights is also of significance. As the true communication pattern is some one-to-one function from Y to X, consider the set X^Y of all m^n functions $f : Y \to X$. By assigning a probability to each ordered pair in the set $X \times Y$, the matrix P ends up inducing a probability on each function in X^Y. The probability that P associates with any $f \in X^Y$ is $\prod\{P_{ij} \mid f(y_j) = x_i\}$, i.e. the weight of the thread in P corresponding to f. By Proposition 1, these weights add up to 1, i.e. we have a probability distribution on the entire set X^Y. If a function f is picked randomly from the set X^Y according to this distribution, i.e. one entry is picked from each column of P according to the probability distribution contained in that column, then the sum of all diagonal weights, $\sum_{d \in \mathcal{D}(P)} \mathcal{W}(d)$, is the probability of the chosen f of being a one-to-one function.

Figure 3(a) shows arbitrarily chosen four of the 24 one-to-one functions that exist in the example of Fig. 2. Their corresponding diagonals in the probability matrix are shown in Fig. 3(b). For example, the function f_1 corresponds to the diagonal $\{(x_1, y_2), (x_2, y_1), (x_4, y_3)\}$, whose weight is $\frac{1}{6} \cdot \frac{1}{2} \cdot \frac{1}{3} = \frac{1}{36}$. The weights of the diagonals corresponding to f_2, f_3, and f_4 in the figure can similarly be seen to be 0, $\frac{1}{18}$, and 0, respectively. In all, this matrix contains 24 diagonals. Of these, the weight of four diagonals is $\frac{1}{18}$, that of 8 diagonals is $\frac{1}{36}$, and that of the remaining 12 diagonals is 0. The sum of all diagonal weights is $\frac{4}{9}$.

As an interesting observation, the set $\mathcal{D}(P)$ of all diagonals of P can be partitioned in a straightforward way using the concept of projections introduced in Bagai et al. [6]. A *projection* is any graph obtained from $K_{m,n}$ by removing some $(m - n)$ vertices from X, and edges connected to those vertices. Clearly, there are $\binom{m}{n}$ projections, each of which has $n!$ diagonals. The collection of all projections results in a convenient partition of $\mathcal{D}(P)$ by grouping together all diagonals embedded in each projection.

Any projection is a complete bipartite graph $K_{n,n}$ and its $n \times n$ matrix Q is obtained from P by removing the corresponding $(m - n)$ rows. The permanent of a square matrix is a well known quantity in combinatorial matrix literature (see for example, Asratian et al. [10]). The *permanent* of any $n \times n$ matrix Q of real numbers is defined as:

$$\text{per}(Q) = \sum_{\pi \in S_n} Q_{1\pi(1)} Q_{2\pi(2)} \cdots Q_{n\pi(n)},$$



(b)

P	y_1	y_2	y_3
x_1	½	⅙	⅙
x_2	½	⅙	⅙
x_3	0	⅓	⅓
x_4	0	⅓	⅓

P	y_1	y_2	y_3
x_1	½	⅙	⅙
x_2	½	⅙	⅙
x_3	0	⅓	⅓
x_4	0	⅓	⅓

P	y_1	y_2	y_3
x_1	½	⅙	⅙
x_2	½	⅙	⅙
x_3	0	⅓	⅓
x_4	0	⅓	⅓

P	y_1	y_2	y_3
x_1	½	⅙	⅙
x_2	½	⅙	⅙
x_3	0	⅓	⅓
x_4	0	⅓	⅓

Fig. 3. (a) Some one-to-one functions in the example of Fig. 2; (b) Diagonals corresponding to these functions.

where S_n is the set of all permutations of the set $\{1, 2, \ldots, n\}$. Let $\mathcal{Q}(P)$ be the set of all $n \times n$ matrices obtained from P by removing some $(m - n)$ rows. The following proposition is immediate.

Proposition 2. *For any probability matrix P,*

$$\sum_{d \in \mathcal{D}(P)} \mathcal{W}(d) = \sum_{Q \in \mathcal{Q}(P)} per(Q).$$

Proof. Immediate from the definition of permanents, and the fact that projections of $K_{m,n}$ induce a partition of $\mathcal{D}(P)$. □

Let $\mathcal{D}(P) = \{d_1, d_2, \ldots, d_{\binom{m}{n}n!}\}$ be the set of diagonals. As stated earlier, if a thread is chosen randomly from P according to the probabilities in P, i.e. one entry from each column of P is chosen according to the probability distribution given by that column, then $\sum_{d \in \mathcal{D}(P)} \mathcal{W}(d)$ is the chance that the chosen thread is a diagonal. However, the attacker already knows that the system's true communication pattern corresponds to a diagonal. Thus, *given* that a diagonal is chosen according to the probabilities in P, a probability is induced on each diagonal of being the chosen one. The *diagonal distribution* induced by P is the normalized sequence of diagonal weights, given by:

$$\phi(P) = \frac{1}{\sum_{d \in \mathcal{D}(P)} \mathcal{W}(d)} \langle\, \mathcal{W}(d_1), \mathcal{W}(d_2), \ldots, \mathcal{W}(d_{\binom{m}{n}n!}) \,\rangle.$$

Clearly, the sum of all values in this sequence is 1, and it is a probability distribution on all diagonals. It captures the probabilities arrived at by the attack on all possible communication patterns of being the actual one employed by the system. The system hopes for values in this sequence to be uniform, as that would give maximum anonymity to its input messages. However, the stronger the attack, the more non-uniform this sequence is. From the attacker's point of view, ideally, only one of the probabilities in it is 1, and all others are 0.

For our example matrix P of Fig. 2, $\phi(P)$ contains four occurrences of the value $\frac{1}{8}$, 8 occurrences of $\frac{1}{16}$, and 12 occurrences of 0.

4 Measuring Anonymity

We now develop a method for measuring the system-wide anonymity that remains after conclusion of a probabilistic attack.

Recall that the system receives m encrypted messages that visibly arrive from some users. Of these, the $(m - n)$ messages that are served from the system's internal cache, thus not sent out, enjoy maximum anonymity as their message content and end-server information remains completely hidden. The remaining n messages are sent out, after decryption, to their destination end-servers. In order to conceal which user sent what message to which end-server, the system attempts to hide its input-output message association, i.e. its true communication pattern. Message encryption/decryption renders impossible any bit-pattern or size comparison between its input and output messages.

The attacker, after analysis, arrives at a probability matrix P that induces a probability distribution on all diagonals of P. The probability associated with any diagonal is that of its being the system's true communication pattern. Given P, the system-wide anonymity remaining can be given by:

$$\Delta(P) = \begin{cases} 1 & \text{if } n = 0, \\ 0 & \text{if } m = n = 1, \\ \frac{1}{m}\left[(m-n) + n\frac{-\sum_{d\in\phi(P)} d \cdot \log(d)}{\log\left(\binom{m}{n} n!\right)}\right] & \text{otherwise.} \end{cases}$$

The value of the above metric always lies between 0 (for no anonymity) and 1 (for full anonymity). If $n = 0$, no output message is sent, thereby giving full anonymity to all input messages. If $m = n = 1$, the sole input message is recognized as the output message, thus has no anonymity. In the general case, the system-wide anonymity is the arithmetic mean of the anonymity received by each of the m input messages. The $(m - n)$ messages served from cache receive full anonymity, i.e. 1. The anonymity of each of the other n messages is computed as the normalized Shannon entropy [11] of the diagonal distribution $\phi(P)$. In this expression, $0 \cdot \log(0)$ is treated as 0. Ever since the works of Diaz et al. [4] and Serjantov et al. [12], employing Shannon entropy of a probability distribution is well accepted for measuring the degree of anonymity upon arrival of that distribution.

For our example matrix P of Fig. 2,

$$\Delta(P) = \frac{1}{4}\left[1 + 3\frac{-4\left(\frac{1}{8}\log(\frac{1}{8})\right) - 8\left(\frac{1}{16}\log(\frac{1}{16})\right)}{\log(24)}\right] \approx 0.823.$$

That is the anonymity enjoyed by its input messages after the attack represented by P.

5 Relation with Existing Anonymity Metrics

Developing ways of precisely measuring the amount of damage occurred by an attack has been of prime importance ever since the first anonymity systems were employed. Much earlier work focussed on measuring anonymity remaining after an attack of a single message output by the system. Chaum [13] proposed simply the number of input message candidates of being that output message as one of the first metrics. Kesdogan et al. [14] improved upon that by proposing as metric the number of input messages that were candidates with a non-zero probability. In their popular works, Serjantov et al. [12] and Diaz et al. [4] were the first to employ Shannon entropy of the probability distribution on all candidates. Tóth et al. [15] favored adopting the maximal probability contained in the distribution as that gave a worst-case measure, which can be more important to users than the average-case. Clauß et al. [16] used the parametric family of entropies of Rényi [17] as the framework for measuring anonymity. Andersson et al. [18] suggested to view the Euclidean distance between the uniform distribution and that arrived at by the attack. Bagai et al. [19] created a graphical framework in which all above metrics were embedded, and proposed a comprehensive usage of certain infinite profiles of the distribution for a more accurate measurement of anonymity.

Instead of focusing on just a single message, Edman et al. [5] gave a method for measuring the *system-wide* anonymity after an attack. However, they considered only infeasibility attacks, which are capable of either rendering an input-output message pair as feasible, or not. The matrix resulting from such an attack is similar to ours, but its entries are limited to be either 0 or 1. Although they attempted to consider probability matrices, like ours, their approach for such matrices was shown by Bagai et al. [7] to be imprecise, who went on to create a better anonymity metric, also based on Shannon entropy. Their method lacked treatment of cache enhanced capabilities of modern anonymity systems.

Bagai et al. [6] were the first to develop an anonymity metric for cache enhanced systems but, like the work of Edman et al. [5], their metric was also limited to infeasibility attacks. Bagai et al. [20], still for infeasibility attacks, then extended this work to measure instead, the anonymity of client-server associations, in which clients and servers send and received multiple messages. As originally argued by Gierlichs et al. [21], such an extension is imperative.

The class of infeasibility attacks is easily seen to be just a finite subclass of the uncountably infinite class of probabilistic attacks. The anonymity measurement technique described in the current paper is for this wider class of attacks on cache enhanced systems.

6 Conclusions and Future Work

We have developed a metric for measuring the degree of anonymity remaining in a cache enhanced anonymity system after a probabilistic attack. Data caching is expected to be prevalent in next-generation anonymity systems, and probabilistic attacks are the most natural and widest class of attacks yet recognized.

While both of these aspects have been considered in existing metrics, our joint treatment of them is novel and needful.

Our approach takes a non-square probability matrix resulting from an attack, and determines the probability distribution induced by it on all possible communication patterns of the system. The extent to which the system's true communication pattern is hidden among these is then measured by taking into account (a) the amount of data caching, and (b) the evenness of this distribution, for which Shannon entropy is employed.

In the process of developing our metric, we noticed some rough indicators that may alternatively be used to reflect the degree of anonymity. For example, given a non-square probability matrix where each column adds up to 1, Shannon entropy can be applied individually to measure the evenness of each column. The purpose is to see how evenly an outgoing message is linked to all incoming messages. The average column evenness can then be used as an anonymity indicator. Although this metric may be argued to be useful, in a way, we feel it to be not the best, as it does not reflect anonymity of communication patterns, and avoids their consideration altogether. Still, comparing our metric with such rough indicators should yield some interesting insights, and we leave that as a direction of future work.

On non-cached systems, probabilistic attacks are well known to be a generalization of infeasibility attacks, in that for any infeasibility attack, there exists a probabilistic attack with the same information, but not *vice versa*. We expect that relationship to hold on cache enhanced systems as well. For cache enhanced systems, Bagai et al. [6] developed a metric for infeasibility attacks, and we expect our metric to be a natural generalization of theirs. A proof of this, however, requires elaborate construction of an information preserving mathematical embedding of non-square infeasibility matrices into probabilistic ones. We have left construction of that embedding, and therefore a proof of generalization as another direction of future work.

References

1. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM **24**(2), 84–88 (1981)
2. Dingledine, R., Mathewson, N., Syverson, P.: Tor: the second-generation onion router. In: Proceedings of the 13th USENIX Security Symposium, pp. 303–320, August 2004
3. Kwon, A., Lazar, D., Devadas, S., Ford, B.: Riffle: an efficient communication system with strong anonymity. Proc. Priv. Enhancing Technol. (PoPETs) **2016**(2), 115–134 (2016)
4. Díaz, C., Seys, S., Claessens, J., Preneel, B.: Towards measuring anonymity. In: Dingledine, R., Syverson, P. (eds.) PET 2002. LNCS, vol. 2482, pp. 54–68. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36467-6_5
5. Edman, M., Sivrikaya, F., Yener, B.: A combinatorial approach to measuring anonymity. In: Proceedings of the IEEE International Conference on Intelligence and Security Informatics, pp. 356–363 (2007)

6. Bagai, R., Tang, B.: Data caching for enhancing anonymity. In: Proceedings of the 25th IEEE International Conference on Advanced Information Networking and Applications, Biopolis, Singapore, pp. 135–142 (2011)
7. Bagai, R., Lu, H., Li, R., Tang, B.: An accurate system-wide anonymity metric for probabilistic attacks. In: Fischer-Hübner, S., Hopper, N. (eds.) PETS 2011. LNCS, vol. 6794, pp. 117–133. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22263-4_7
8. Shubina, A.M., Smith, S.W.: Using caching for browsing anonymity. ACM SIGEcom Exchanges **4**(2), 11–20 (2003)
9. Kim, B.R., Kim, K.C.: Efficient caching strategies for gnutella-like systems to achieve anonymity in unstructured P2P file sharing. In: Etzion, O., Kuflik, T., Motro, A. (eds.) NGITS 2006. LNCS, vol. 4032, pp. 117–128. Springer, Heidelberg (2006). https://doi.org/10.1007/11780991_11
10. Asratian, A., Denley, T., Häggkvist, R.: Bipartite Graphs and Their Applications. Cambridge University Press, Cambridge (1998)
11. Shannon, C.: A mathematical theory of communication. Bell Syst. Techn. J. **27**, 379–423, 623–656 (1948)
12. Serjantov, A., Danezis, G.: Towards an information theoretic metric for anonymity. In: Dingledine, R., Syverson, P. (eds.) PET 2002. LNCS, vol. 2482, pp. 41–53. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36467-6_4
13. Chaum, D.: The dining cryptographers problem: unconditional sender and recepient untraceability. J. Cryptol. **1**, 65–75 (1988)
14. Kesdogan, D., Egner, J., Büschkes, R.: Stop- and- Go-MIXes providing probabilistic anonymity in an open system. In: Aucsmith, D. (ed.) IH 1998. LNCS, vol. 1525, pp. 83–98. Springer, Heidelberg (1998). https://doi.org/10.1007/3-540-49380-8_7
15. Tóth, G., Hornák, Z., Vajda, F.: Measuring anonymity revisited. In: Proceedings of the 9th Nordic Workshop on Secure IT Systems, Espoo, Finland, pp. 85–90 (2004)
16. Clauß, S., Schiffner, S.: Structuring anonymity metrics. In: Proceedings of the ACM Workshop on Digital Identity Management, pp. 55–62 (2006)
17. Rényi, A.: On measures of entropy and information. In: Proceedings of the 4th Berkeley Symposium on Mathematical Statistics and Probability, pp. 547–561 (1961)
18. Andersson, C., Lundin, R.: On the fundamentals of anonymity metrics. In: Fischer-Hübner, S., Duquenoy, P., Zuccato, A., Martucci, L. (eds.) Privacy and Identity 2007. ITIFIP, vol. 262, pp. 325–341. Springer, Boston, MA (2008). https://doi.org/10.1007/978-0-387-79026-8_23
19. Bagai, R., Jiang, N.: Measuring anonymity by profiling probability distributions. In: Proceedings of the 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TRUSTCOM), Liverpool, UK, pp. 366–374 (2012)
20. Bagai, R., Lu, H.: Measuring client-server anonymity. In: Gaj, P., Kwiecień, A., Stera, P. (eds.) CN 2016. CCIS, vol. 608, pp. 96–106. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39207-3_9
21. Gierlichs, B., Troncoso, C., Diaz, C., Preneel, B., Verbauwhede, I.: Revisiting a combinatorial approach toward measuring anonymity. In: Proceedings of the 7th ACM Workshop on Privacy in the Electronic Society, Alexandria, VA, USA, pp. 111–116 (2008)

The Impact of Time Parameters on the Security Protocols Correctness

Sabina Szymoniak(✉)

Institute of Computer and Information Sciences,
Czestochowa University of Technology, Dabrowskiego 73,
42-200 Czestochowa, Poland
sabina.szymoniak@icis.pcz.pl

Abstract. The security of computer network users is a very important aspect of the network administrators' work. The related analysis and verification of security protocols (SP) is a key tool in the fight against access to data by unauthorized users.

This paper will present analysis and verification of timed security protocols. In order to carry out research the formal model and the computational structure presented in earlier works, were extended. Protocols' analysis was related to the influence of time parameters on its security. The times of message encryption and decryption and delays in the network were investigated. The tests were carried out in two stages. In the first stage, the possibility to carry out generated protocols was tested. In the second stage, the execution times were analyzed for different values of time parameters. We proofed that the proper analysis and appropriately adopted time conditions make the protocol safer.

Keywords: Security protocols · Modeling & verification
Time analysis

1 Introduction

Security is one of the most important aspect of Internet communication. In order to ensure appropriate security level a special security protocols are designed. Security protocol (SP) consist of several steps which are used to users' authentication and to ensure the confidentiality of data. In such communication, the time aspect is decisive. Just like in real life, fractions of seconds decide about our to be or not to be in this world. Each passing second during Internet communication may result in the loss of our data. Unfortunately, even the best security systems will not guarantee security if they are not regularly verified. Badly chosen safety rules, including time constraints, may allow unauthorized users access to confidential information.

So far, many research teams have been researching the issues related to the security protocols verification. These studies concerned various properties and aspects of security protocols [1–4]. These methods also can be classified according

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 333–343, 2018.
https://doi.org/10.1007/978-3-319-92459-5_27

to several criteria [5,6]. Many approaches have been developed, using various techniques or mathematical models [7–9]. The indicated studies are aimed at examining the correctness of SP. It is necessary to obtain information whether the proposed protocol satisfied the goals.

An interesting methodology of SP modeling was proposed by Jakubowska and Penczek [10–12]. In this approach timed and untimed protocols were included. Their method consisted in calculating the correct execution time of SP. Jakubowska and Penczek checked if the Intruder could modify this time by attacking. The protocol execution model included timeouts and delays in the network. In turn, time constraints allowed to indicate the impact of time on the security of the protocol. In this approach, the research concerned only a single session, the interlaces of protocol's executions were not considered.

The presented method of SP verification takes into account the impact of various time aspects on their safety. Properly selected values of these parameters allow to indicate the limitations that should be applied for safe communication. The conducted research confirms a significant impact of time on the data security sent via electronic links.

The rest of the paper is organized as follows. The next section describes the BAN concrete Andrew Secure RPC protocol. This protocol will be used to discuss the research results. The Sect. 3 describes the research methodology. All assumptions regarding time parameters have been described. In the next section, we present the experimental results obtained during the research. The last section presents our conclusions.

2 BAN Concrete Andrew Secure RPC

One of the security protocols is BAN concrete Andrew Secure RPC, Andrew protocol for short. This protocol was proposed in 1989 by Michael Burrows, Abadi and Needham in [13]. Andrew protocol uses symmetric cryptography to exchange a new shared key. We created timed version of this protocol by replacing nonces by timestamps. Syntax of timed Andrew protocol version in Common Language is as follows:

$$\begin{aligned}
&\alpha_1\ A \rightarrow B : A, T_A,\\
&\alpha_2\ B \rightarrow a : \{T_A, K'_{AB}\}_{K_{AB}},\\
&\alpha_3\ A \rightarrow B : \{T_A\}_{K'_{AB}},\\
&\alpha_4\ B \rightarrow A : T_B.
\end{aligned}$$

In Andrew protocol occurs two user, A and B. Initially, the symmetric key K_{AB} is known only to this two users. This key is shared between them. In first step user A sends its timestamp T_A with its ID to user B. In response, user B generates new session key K'_{AB} and sends it to A with timestamp T_A encrypted by shared key K_{AB}. In third step, user A sends to user B its timestamp T_A, encrypted by new shared key. At least, in the last step, user B generates timestamp T_B and send it to A. Symmetric key K'_{AB} will be used also in the future session.

3 Research Methodology

In order to carry out the research, a special formal model and a computational structure were prepared.

The formal model allows to define the timed protocol as an algorithm. In this model, definitions of sets of time conditions, step and protocol (set of steps) are presented. In set of time conditions delays in the network were included. The time protocol step takes into account external and internal actions performed during the protocol and imposed time conditions. Formal definitions allow full specification of step and the whole protocol. Thanks to these definitions, it is possible to specify timed and untimed protocols. These definitions were described in more details in [14,15].

The computational structure allows defining protocol's executions including the Intruder. Thanks to this, it will be possible to consider the protocols executions in the real network.

In the computational structure interpretations of the protocol, timed protocol's step, user's knowledge, protocol's calculation and time dependencies have been defined. Interpretations determine a set of different executions of the examined protocol. These executions differ from each other in their execution time. The interpreting function makes it possible to map timestamps, message sending times and delays in the network into non-negative real numbers. In this way, one protocol, which is specific in time, is obtained. This function is used during defining a step and calculation of the timed protocol.

Timed protocol's step is defined by two tuples (α^1, α^2), where:

$$\begin{aligned} \alpha^1 &= (S_{\rightarrow}; R_{\leftarrow}; L), \\ \alpha^2 &= (\tau; D; X; G; tc). \end{aligned}$$

In this notation, we marked sender by $S_{\rightarrow}$ and receiver by $R_{\leftarrow}$. L denote the message being sent in a given step. This tuple responds to protocol's syntax in Common Language. In the second tuple by τ we denote the time which is necessary to send the message, by D denote delay in the network, X means the collection of letters. From this letters message L will be constructed. G is a collection of letters that must be generated by the sender for the purpose of constructing a message. tc is a set of timed conditions. This conditions should be fulfilled to allow the execution of the protocol.

As an example, the formal definition of the Andrew protocol is presented:

- $\alpha_1 = (\alpha_1^1, \alpha_1^2)$:
  - $\alpha_1^1 = (A; B; I_A, \tau_A)$,
  - $\alpha_1^2 = (\tau_1; D_1; \{I_A, \tau_B\}; \{\tau_B\}; \tau_1 + D_1 \leq \mathcal{L}_{\mathcal{F}})$.
- $\alpha_2 = (\alpha_2^1, \alpha_2^2)$:
  - $\alpha_2^1 = (B; A; \langle \tau_B, K'_{AB} \rangle_{K_{AB}})$,
  - $\alpha_2^2 = (\tau_2; D_2; \{\tau_B, K_{AB}, K'_{AB}\}; \{K'_{AB}\}; \tau_2 + D_2 - \tau_A \leq \mathcal{L}_{\mathcal{F}})$.

– $\alpha_3 = (\alpha_3^1, \alpha_3^2)$:

  - $\alpha_3^1 = (A; B; \langle \tau_A \rangle_{K'_{AB}})$,
  - $\alpha_3^2 = (\tau_3; D_3; \{\tau_B, K'_{AB}\}, \{\emptyset\}, \tau_3 + D_3 - \tau_A \leq \mathcal{L}_{\mathcal{F}})$.

– $\alpha_4 = (\alpha_4^1, \alpha_4^2)$:

  - $\alpha_4^1 = (B; A; \tau_B)$,
  - $\alpha_4^2 = (\tau_4; D_4; \{\tau_B\}; \{\tau_B\}; \tau_4 + D_4 - \tau_B \leq \mathcal{L}_{\mathcal{F}})$.

In first step of Andrew protocol user A sents to user B a message I_A, τ_A. In order to construct this message the τ_A is needed. User A must generate this object. The message sending time is increased by the delay in the network and reduced by the value of the timestamp τ_A. This time must be less or equal to the assumed lifetime. The following steps in Andrew protocol should be considered similarly. Please note that the notation $\langle \tau_B, K'_{AB} \rangle_{K_{AB}}$ (in second step) means that tau_B and K'_{AB} were encrypted by symmetric key K_{AB}, which is shared between users A and B.

Knowledge operators define the methods of knowledge acquiring and using by honest users and by the Intruder. During the execution of the protocol, the so-called initial knowledge of each user was included. Initial knowledge is associated with elements that are publicly available and shared with other users (in the case of honest users), as well as elements previously generated (in the case of Intruder).

The most important element of the computational structure is the definition of time dependencies. For the needs of the studies carried out in [15,16], time dependencies related to the time of composing the message, duration of the step, duration of the step (minimum, current, maximum), session duration (minimum, current, maximum) and lifetime were defined.

Using defined relationships the research was conducted. This research showed that the value of delay in the network has a huge impact on the correctness of protocol's executions and the intruder's capabilities. If the time constraints values are badly chosen, Intruder can acquire a range of attack options. Therefore, it was decided to determine the dependencies for other communication time parameters and check their impact on the correctness of protocol's executions.

In order to determine time dependencies the following symbols have been defined:

– n - number of protocol steps,
– k - protocol step number,
– T_e - current encryption time,
– T_e^{min} - minimum encryption time,
– T_e^{max} - maximum encryption time,
– T_d - current decryption time,
– T_d^{min} - minimum decryption time,
– T_d^{max} - maximum decryption time,

- T_g - time of generating confidential information for sending messages,
- D - current delay in the network,
- D_{min} - minimum delay in the network,
- D_{max} - maximum delay in the network,
- T_c - time of composing the message,
- T_k - step time,
- T_k^{min} - minimum step time,
- T_k^{max} - maximum step time,
- T_{ses} - session time (the sum of times of all protocol steps),
- T_{ses}^{min} - minimum session time,
- T_{ses}^{max} - maximum session time,
- T_{out}^{k} - lifetime in step k-th.

The encryption time is the time that user needs to encrypt the message and the decryption time is the time that user needs to decrypt the message. We differentiate the current, minimum and maximum time of encryption and decryption. This distinction is necessary because the time needed to perform these actions depends on the power of the computer unit on which the calculations are made. Therefore, it should be assumed the time range in which these operations should take place in one step.

The situation is similar in the case of delays in the network, which also should be taken and consider the range of these values. The transmission time is dependent on the parameters of the network, so it is necessary to consider the different values of this parameter.

Step times are calculated according to the following formulas:

$$T_k = T_e + T_g + D + T_d \tag{1}$$

$$T_k^{min} = T_e^{min} + T_g + D_{min} + T_d^{min} \tag{2}$$

$$T_k^{max} = T_e^{max} + T_g + D_{max} + T_d^{max} \tag{3}$$

Of course session times are related to the corresponding values of the step times. Lifetime in the k-th step indicates the maximum waiting time for a response. These dependencies were calculated according to the formulas presented in [15].

The research on security protocols has been carried out in two stages. In the first stage the possibility of performing the generated executions using the SAT-solver MiniSAT was checked. Next, the analysis of the duration of the protocols executions was made. During the analysis of the times, various time aspects were considered. The time parameters values will be described in more detail in the next section. All tests were carried out using an abstract time unit ([tu]). This unit is an arbitrarily selected and considered time period.

The tests were performed using a computer unit with the Linux Ubuntu operating system, processor Intel Core i7 and 16 GB RAM.

4 Experimental Results

Andrew protocol research began with the generation of all protocol's executions using the implemented tool. Some aspects of this tool were described in [15,16]. A summary of the executions is presented in Table 1.

Table 1. Summary of Andrew protocol's executions

No.	Participants	Parameters	No.	Participants	Parameters
1	A→B		10	B→A	
2	I→B	T_I, K_{IB}	11	I→A	T_I, K_{IA}
3	I→B	T_A, K_{IB}	12	I→A	T_B, K_{IA}
4	I(A)→B	T_I, K_{AB}	13	I(B)→A	T_I, K_{AB}
5	I(A)→B	T_A, K_{AB}	14	I(B)→A	T_B, K_{AB}
6	A→I	T_I, K_{IA}	15	B→I	T_I, K_{IB}
7	A→I	T_B, K_{IA}	16	B→I	T_A, K_{IB}
8	A→I(B)	T_I, K_{AB}	17	B→I(A)	T_I, K_{AB}
9	A→I(B)	T_B, K_{AB}	18	B→I(A)	T_A, K_{AB}

The table contains information about the execution number, execution participants and parameters which are used by the Intruder during communication. The executions were numbered to simplify appealing to them. The execution's participants are presented in the order in which they appear during a given execution. Designations A and B point to honest users. The signs I, $I(A)$, and $I(B)$ indicate the Intruder (I) and the Intruder impersonating honest users ($I(A)$, $I(B)$).

Then, these executions were checked using the SAT-solver MiniSAT. The SAT-solver indicated the executions of no. 4 and 13 as impossible to carry out. As the attacking execution the interlace of the 8 and 5 executions was marked. The scheme of this interlace is as follows:

8.1 $A \rightarrow I(B) : I_A, T_A$
- 5.1 $I(A) \rightarrow B : I_A, T_A$
- 5.2 $B \rightarrow I(A) : \{T_A, K'_{AB}\}_{K_{AB}}$

8.2 $I(B) \rightarrow A : \{T_A, K'_{AB}\}_{K_{AB}}$

8.3 $A \rightarrow I(B) : \{T_A\}_{K'_{AB}}$
- 5.3 $I(A) \rightarrow B : \{T_A\}_{K'_{AB}}$
- 5.4 $B \rightarrow I(A) : T_B$

8.4 $I(B) \rightarrow A : T_B$

User A tries to connect with user B. However, the message from step 8.1 goes to the Intruder who impersonates the user B. Intruder does not have adequate knowledge to execute step 8.2. Therefore, he must take additional steps from execution 5 in order to acquire relevant knowledge. The presented execution is

a man in the middle attack. This interlace will be used to discuss the research results later in this section.

For the tests following assumptions have been made:

- $T_g = 1[tu]$,
- $T_e^{min} = T_d^{min} = 1[tu]$,
- $T_e^{max} = T_d^{max} = 5[tu]$,
- $D_{min} = 1[tu]$,
- $D_{max} = 3[tu]$.

Then, for such assumptions in accordance with the methodology of [15], the values of the minimum and maximum session time were determined:

- $T_{ses}^{min} = 11[tu]$
- $T_{ses}^{max} = 35[tu]$

Next, lifetime values in individual steps were determined:

- $T_{out}^1 - 35[tu]$,
- $T_{out}^2 - 32[tu]$,
- $T_{out}^3 - 17[tu]$,
- $T_{out}^4 - 4[tu]$.

The next step in the study of the Andrew protocol was the analysis of session times of individual executions. In this study different values of encryption and decryption time and delay in the network were used. First, the situation in which both the time of encryption and the delay in the network assumed values equal to the lower limit of the adopted range of these values were checked.

Table 2. Timed analysis of execution no. 8

Basic step	Additional step	T_e	T_g	D	T_d	T_k	Comment
8.1		0	1	1	0	2	Ok
	5.1	0	0	1	0	(1)	Ok
	5.2	1	1	1	0	(3)	Ok
8.2		0	0	1	1	6	Ok
8.3		1	0	1	0	2	Ok
	5.3	0	0	1	1	(2)	Ok
	5.4	0	1	1	0	(2)	Ok
8.4		0	0	1	0	5	$!T_{out}^4$

The Table 2 presents the analysis of session time of execution no. 8 including additional steps time. In brackets additional step times, which are included in basic times, have been marked. The first step (8.1) executed within 2 [tu]. The imposed time condition has been preserved. The second step time (8.2) took into

account the additional steps (5.1 and 5.2). This step lasted 6 [tu]. Again, the imposed time condition has been preserved. Step 8.3 executed within 2 [tu], while preserving the time condition. Step 8.4 also took into account the additional steps times (5.3 and 5.4). The total time of this step was 5 [tu]. The time condition imposed on the fourth step was not met. Communication should therefore be interrupted.

The times analysis for the values $T_e = T_d = 1[tu]$ and $D = 1[tu]$ showed the influence of time parameters values on session time and correctness of protocol execution.

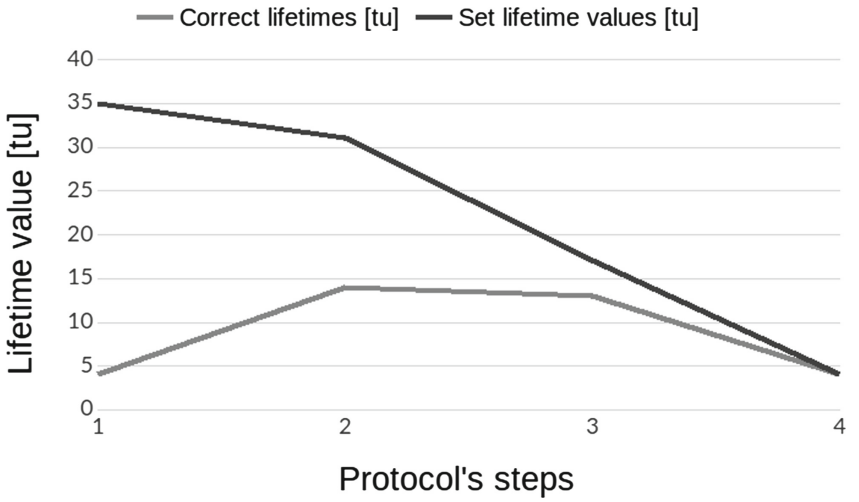


Fig. 1. Dependence of set and correct lifetime values for Andrew protocol

The protocol structure includes steps during which no encryption and decryption operations were performed (steps 1 and 4). In addition, due to the structure of the attack, the Intruder also did not perform encryption and decryption operations. Despite not counting the duration of these operations, the Intruder was not able to carry out a successful attack. Unfulfilled time condition in the fourth step prevented the attack.

The implemented tool also suggested which values should be taken in lifetime in individual steps to keep the protocol safe. The values were obtained by analyzing the steps structure and carried out tests. The new lifetime values are as follows:

- $T^1_{out} = 4[tu]$,
- $T^2_{out} = 14[tu]$,
- $T^3_{out} = 13[tu]$,
- $T^4_{out} = 4[tu]$.

Figure 1 shows the dependence of set and correct lifetime values. Only for the fourth step the set and correct values are equal. The presented results show the correct lifetime values in protocol steps. These values should be set to avoid Intruder's attack.

In the further part of the research, a similar analysis of executions times using other values of time parameters was carried out.

Table 3. Session times of execution no. 8 for different encryption and decryption times

Steps	Session times [tu]				
	$T_e = T_d = 1[tu]$	$T_e = T_d = 2[tu]$	$T_e = T_d = 3[tu]$	$T_e = T_d = 4[tu]$	$T_e = T_d = 5[tu]$
1	4	4	4	4	4
2	12	14	16	18	20
3	4	5	6	7	8
4	11	12	13	14	15

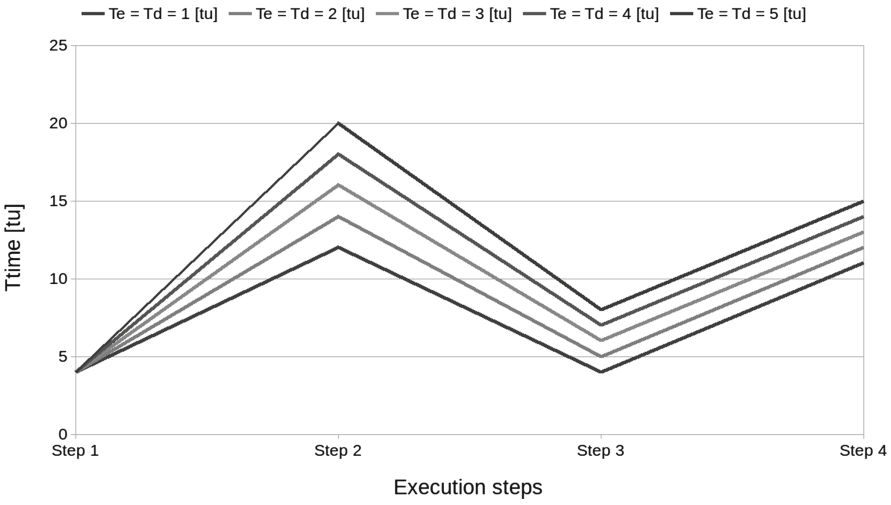


Fig. 2. Step time values for different encryption and decryption times

The Table 3 presents the values of session times (taking into account additional steps times). Encryption and decryption times took values from 1 to 5 [tu], generation time was equal 1[tu] and current delay in the network value was equal 3 [tu].

It can be observed that for every consider situation the time condition imposed on step 4 was not met. A very interesting situation arose for encryption times equal to 3, 4 and 5 [tu]. If the communication will be not interrupted in the

fourth step as a result of exceeding the imposed time condition, then subsequent time units would exceed the maximum allowable duration of the session. In the last analyzed case ($T_e = 5[tu]$), the maximum session time will be exceeded first. The time of the first three steps is equal 32 [tu]. Next three units, added during the fourth step, will exceed T_{ses}^{max}.

The Fig. 2 shows summary of step times values including additional steps. The lines marked on the graph refer to the respective test series and the used values of encryption and decryption times in these series (from 1 to 5 [tu]). These values show the dependence between a particular series of tests.

5 Conclusion

In this paper was presented analysis and verification of the Andrew protocol's timed version. The analysis was related to the influence of time parameters on protocol's security. The tests took into account the encryption and decryption times and delays in the network. In order to carry out the research, the formal model and the computational structure proposed in [14,15] were used and extended. The calculations were made using the implemented tool and SAT-Solver MiniSAT. The tests were carried out in two stages. In the first stage, the possibility of performing generated executions of Andrew protocol was checked. In the second stage, the execution times were analyzed for different values of time parameters (encrypted and decrypted times, delays in the network).

The obtained results showed the impact of time parameters on the protocol's executions correctness and users' security. Too lower maximum value of delay in the network may result that honest users will not be able to execute protocol in correct time. Too higher maximum value of delay in the network may allow Intruder to carry out an attack.

Incorrectly selected values of time parameters give the Intruder a range of attack capabilities. Therefore, it is necessary to verify computer network's operation on a regular basis. Due to proper analysis and appropriately adopted lifetime restrictions, the protocol will become safer.

Our further research will be related to the computer network's simulation taking into account the random encryption and decryption times values. These values will be generated with different probability distribution.

References

1. Cortier, V., Delaune, S., Lafourcade, P.: A survey of algebraic properties used in cryptographic protocols. J. Comput. Secur. **14**(1), 1–43 (2006)
2. Bolignano, D.: An approach to the formal verification of cryptographic protocols. In: Third ACM Conference on Computer and Communications Security, pp. 106–118. ACM Press (1996)
3. Satyanarayanan, M.: Integrating security in a large distributed system. ACM Trans. Comput. Syst. **7**(3), 247–280 (1989)
4. Sprenger, C., Basin, D.: Refining security protocols. J. Comput. Secur. **26**(1), 71–120 (2018)

5. Meadows, C.A., Meadows, C.A.: Formal verification of cryptographic protocols: a survey. In: Pieprzyk, J., Safavi-Naini, R. (eds.) ASIACRYPT 1994. LNCS, vol. 917, pp. 133–150. Springer, Heidelberg (1995). https://doi.org/10.1007/BFb0000430
6. Kurkowski, M.: Formalne metody weryfikacji wlasnosci protokolow zabezpieczających w sieciach komputerowych. Informatyka - Akademicka Oficyna Wydawnicza EXIT (2013)
7. Corin, R., Etalle, S., Hartel, P.H., Mader, A.: Timed analysis of security protocols. J. Comput. Secur. **15**(6) (2007)
8. Basin, D., Modersheim, S., Viganò, L.: Technical report no. 450 OFMC: a symbolic model-checker for security protocols (2004)
9. Cremers, C., Mauw, S.: Operational Semantics and Verification of Security Protocols. Information Security and Cryptography. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-540-78636-8
10. Jakubowska, G., Penczek, W., Srebrny, M.: Verifying timed security protocols via translation to timed automata. In: International Workshop on Concurrency, Specification and Programming (CS&P 2002). Lecture Notes in Computer Science, pp. 100–115 (2005). Warsaw University
11. Jakubowska, G., Penczek, W.: Is your security protocol on time? In: Arbab, F., Sirjani, M. (eds.) FSEN 2007. LNCS, vol. 4767, pp. 65–80. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75698-9_5
12. Jakubowska, G., Penczek, W.: Modelling and checking timed authentication of security protocols. Fundam. Inform. **79**(3–4), 363–378 (2007)
13. Burrows, M., Abadi, M., Needham, R.: A logic of authentication. ACM Trans. Comput. Syst. **8**(1), 18–36 (1990)
14. Szymoniak, S., Kurkowski, M., Piatkowski, J.: Timed models of security protocols including delays in the network. J. Appl. Mathe. Comput. Mech. **14**(3), 127–139 (2015)
15. Szymoniak, S., Siedlecka-Lamch, O., Kurkowski, M.: Timed analysis of security protocols. In: Grzech, A., Świątek, J., Wilimowska, Z., Borzemski, L. (eds.) Information Systems Architecture and Technology: Proceedings of 37th International Conference on Information Systems Architecture and Technology – ISAT 2016 – Part II. AISC, vol. 522, pp. 53–63. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-46586-9_5
16. Szymoniak, S.: Analysis and simulations of timed security protocols. Studa Inf. **38**(2), 55–66 (2017)

On Some Time Aspects in Security Protocols Analysis

Sabina Szymoniak[1(✉)], Olga Siedlecka-Lamch[1], and Mirosław Kurkowski[2]

[1] Institute of Computer and Information Sciences, Czestochowa University of Technology, Dabrowskiego 73, 42-200 Czestochowa, Poland
{sabina.szymoniak,olga.siedlecka}@icis.pcz.pl
[2] Institute of Computer Science, Cardinal St. Wyszyński University, Wóycickiego 1/3, 01-938 Warsaw, Poland
m.kurkowski@uksw.edu.pl

Abstract. In many verification approaches for security protocols analysis time aspects are omitted. According to this in our work we try to show how these problems are important in this area. To do this we present new ideas as well as methods for calculating and checking several types of time parameters that characterize some time aspects during and after the protocol's execution. As an example we present the timed analysis in the case of the timed version of the well known the NSPKL protocol (Needham Schroeder Public Key Protocol revised by Lowe). The experimental results obtained using a proprietary tool are also shown. Using this, during the running of the protocol, the "presence" of the Intruder can be followed by observing incorrect time of the protocol execution. As we will see, both those too short and too long allows this.

Keywords: Security protocols · Modeling & verification
Time analysis

1 Introduction

For a few decades, one of the most important aspects of modern information systems has been their security which would not be possible without specially designed security protocols, which are some kind of communication protocols widely used in computer networks or systems. These protocols guarantee the correct authorization and confidentiality of transmitted data. Even though they are just one of the smallest pieces of a code of large communication systems, we can say that these protocols are often the key points in computer systems security. The importance of using these tools and their correctness is significant. Sometimes all it takes is a small error combined with the conscious operation of the broadly understood Intruder for the user to loose valuable data.

In order to avoid errors, for years various formal methods for correctness verification and the analysis of security protocols have been introduced and practically used. In the case of formal ways we can find the inductive [12], deductive [3],

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 344–356, 2018.
https://doi.org/10.1007/978-3-319-92459-5_28

model checking [10], and other methods [5,14] among them. These methods allowed the detection of many possibilities of attacks upon protocols, also in the case of protocols that were fully used on the market and were considered safe. Practice has begun behind formal methods and many projects as well as tools have been implemented to study the security of protocols. The most important ones include: ProVerif [2], Scyther [4], and AVISPA [1].

In none of the aforementioned tools, time aspects of protocols executions are the subject of research. Intuitively, many designers added timestamps to enhance security, but it has not been studied how time parameters should be selected to be able to register the presence of the Intruder. One of the first attempts of such analysis appeared in works of Jakubowska and Penczek [7], [6]. The combination of observations done by Jakubowska and the formal model introduced in Kurkowski's works [8,9] created a good basis for adding time parameters into research in this area. These parameters are related to encryption, decryption, and information generation time. Thus, we can create and consider a model that allows detailed time analysis for security protocols executions.

We will develop our considerations on the example of the well known the NSPKL protocol, which can be an excellent example for investigation. This protocol was described in 1995 by Lowe in paper [10]. It was a response to the original NSPK protocol - designed by Needham and Schroeder in the seventies of the last century [11] - as it turned out, the protocol was vulnerable to attacks resulting from interlaced executions. The Lowe's modification had to prevent attacks. This protocol uses asymmetric cryptography for mutual authentication.

Let us consider a timed version of this protocol obtained by replacing nonces with timestamps. The syntax of the NSPKL protocol's timed version written in Common Language is as follows:

$$\begin{aligned}
&\alpha_1 \quad A \rightarrow B : \langle T_A \cdot I_A \rangle_{K_B}, \\
&\alpha_2 \quad B \rightarrow A : \langle T_A \cdot T_B \cdot I_B \rangle_{K_A}, \\
&\alpha_3 \quad A \rightarrow B : \langle T_B \rangle_{K_B}.
\end{aligned}$$

In the NSPKL protocol two users occur: A and B. In the first step, A generates timestamp T_A and sends it with his ID encrypted by the public key of user B. In the next step, user B generates timestamp T_B and sends it to A with timestamp T_A and B's ID encrypted by the public key of user A. In the last step, user A sends to user B his timestamp T_B, encrypted by K_B.

So far this protocol has been treated as safe in that sense that there have not been a single execution or interleaved several executions that allow cheating an honest user by a malicious Intruder in the case of the users' authentication, and the secret data transmitted during protocol's executions cannot be compromised.

As we know from the practical point of view, an another important problem is the correctness of using protocols from the time point of view. In real computer networks there is a need that a communication session and a protocol included in it should be executed in a precisely specified period of time. This was one of reasons for introducing into protocol's schemes time tickets (timestamps) that allows time control by users during the network operation.

As we mentioned before, problems connected with the time restriction of protocol executions in real networks have been not satisfactory considered before. Some ideas introduced and presented in works of Jakubowska and Penczek have been not developed.

The rest of the paper is organized as follows. In the next Section a fragment of the formal model of protocols' executions that allows the analysis regarding time parameters will be shown. We will describe several types of time parameters that characterize some important time aspects during and after the protocol's execution. Next, we will show experimental results for the timed version of the NSPKL protocol obtained using our author's tool [15]. Finally, we will present conclusions and plans for future work.

2 Formal Model and Computational Structure

In our research we use the automatic protocols' verification using a specially designed tool. For this purpose a suitable specification of the protocol is needed. This depends on some formal notations and structures that allows expressing all what is needed and important from the verification's goals point of view.

First, a formal definition of protocol steps is required. Every step of a timed protocol (taking into account network delays) is defined by two tuples α^1, α^2:

$$\alpha^1 = (S_{\rightarrow}, R_{\leftarrow}, L), \qquad \alpha^2 = (\tau, D, X, G, tc). \tag{1}$$

In this notation the first tuple consists of information similar to the previous method used for protocol's specification. It contains the sender, the receiver, and the scheme of message sent respectively.

The second tuple includes:

- τ denotes the moment when the message was sent,
- D denotes the value of network delay,
- X is the set of letters necessary to construct the message,
- G is the set of letters that must be generated by the sender in order to create the message.
- tc is the set of temporal conditions that should be fulfilled to enable the protocol execution.

Observe that the second tuple consists additionally of time aspects that characterize proper time conditions that should be achieved in the protocol's execution.

In order to carry out the research, a formal model and a computational structure from [9] were extended. In the aforementioned book it was possible to define a timed and untimed protocol as an abstract algorithm. In the protocol's definition time conditions were included. Unfortunately, the authors did not consider delays in the network. The delays in the network were added for the current research. A new time conditions' set is defined by the following grammar:

$$\mathbf{tc} :: = true \mid \tau_i + \tau_d - \tau_j \leq \mathcal{L}_{\mathcal{F}} \mid tc \wedge tc, \tag{2}$$

where: τ_i is a time moment, τ_d is a delay, τ_j is a value of time ticket, and $\mathcal{L}_{\mathcal{F}}$ is the lifetime.

For understanding our formalizations we present, as an example, a formal definition of the timed version of the NSPKL Protocol. The protocol consists of three steps: $\alpha_1, \alpha_2, \alpha_3$, that can we written in the computational structure in the following way:

$$\begin{aligned} &\alpha_1 = (\alpha_1^1, \alpha_1^2), \\ &\alpha_1^1 = (\mathcal{A}; \mathcal{B}; \langle \tau_{\mathcal{A}} \cdot \mathcal{I}_{\mathcal{A}} \rangle_{\mathcal{K}_B}), \\ &\alpha_1^2 = (\tau_1; \mathcal{D}_1; \{\tau_{\mathcal{A}}, \mathcal{I}_{\mathcal{A}}, \mathcal{K}_{\mathcal{B}}\}; \{\tau_{\mathcal{A}}\}; \tau_1 + \mathcal{D}_1 - \tau_{\mathcal{A}} \leq \mathcal{L}_f). \end{aligned} \tag{3}$$

In the first step $\mathcal{A}$ and $\mathcal{B}$ denote a sender and a receiver, respectively. $\mathcal{I}_{\mathcal{A}}$ is the identifier of the user $\mathcal{A}$. $\tau_{\mathcal{A}}$ is the time ticket generated by the user $\mathcal{A}$. $\langle \tau_{\mathcal{A}} \cdot \mathcal{I}_{\mathcal{A}} \rangle_{\mathcal{K}_B}$ is the message sent in this step. The tuple α_1^2 includes the following temporal aspects - τ_1 denotes the moment when the message was sent, $\mathcal{D}_1$ denotes the delay of the network, $\{\tau_{\mathcal{A}}, \mathcal{I}_{\mathcal{A}}, \mathcal{K}_{\mathcal{B}}\}$ is the set of letters necessary to construct the message $\langle \tau_{\mathcal{A}} \cdot \mathcal{I}_{\mathcal{A}} \rangle_{\mathcal{K}_B}$. In this set we distinguish the set $\{\tau_{\mathcal{A}}\}$ containing the letter that must be generated by the sender $\mathcal{A}$ in order to create the message $\langle \tau_{\mathcal{A}} \cdot \mathcal{I}_{\mathcal{A}} \rangle_{\mathcal{K}_B}$. $\tau_1 + \mathcal{D}_1 - \tau_{\mathcal{A}} \leq \mathcal{L}_f$ is a temporal condition that should be fulfilled to enable the protocol execution. In this case the condition states that the value of the message sending time $(\tau_1 - \tau_{\mathcal{A}})$ plus value of the network's delay $\mathcal{D}_1$ should be less or equal to the lifetime $\mathcal{L}_f$.

$$\begin{aligned} &\alpha_2 = (\alpha_2^1, \alpha_2^2), \\ &\alpha_2^1 = (\mathcal{B}; \mathcal{A}; \langle \tau_{\mathcal{A}} \cdot \tau_{\mathcal{B}} \cdot \mathcal{I}_{\mathcal{B}} \rangle_{\mathcal{K}_{\mathcal{A}}}), \\ &\alpha_2^2 = (\tau_2; \mathcal{D}_2; \{\tau_{\mathcal{A}}, \tau_{\mathcal{B}}, \mathcal{I}_{\mathcal{B}}, \mathcal{K}_{\mathcal{A}}\}; \{\tau_{\mathcal{B}}\}; \tau_2 + \mathcal{D}_2 - \tau_{\mathcal{A}} \leq \mathcal{L}_f \wedge \tau_2 + \mathcal{D}_2 - \tau_{\mathcal{B}} \leq \mathcal{L}_f). \end{aligned} \tag{4}$$

In the second step $\mathcal{B}$ and $\mathcal{A}$ denote the sender and the receiver again. $\tau_{\mathcal{B}}$ is the time ticket generated by the user $\mathcal{B}$. $\langle \tau_{\mathcal{A}} \cdot \tau_{\mathcal{B}} \cdot \mathcal{I}_{\mathcal{B}} \rangle_{\mathcal{K}_{\mathcal{A}}}$ is the message sent in this step. τ_2 denotes the moment when the message was sent, $\mathcal{D}_2$ denotes the delay of the network, $\{\tau_{\mathcal{A}}, \tau_{\mathcal{B}}, \mathcal{I}_{\mathcal{B}}, \mathcal{K}_{\mathcal{A}}\}$ is the set of letters necessary to construct the message $\langle \tau_{\mathcal{A}} \cdot \tau_{\mathcal{B}} \cdot \mathcal{I}_{\mathcal{B}} \rangle_{\mathcal{K}_A}$. The set $\{\tau_{\mathcal{B}}\}$ contains the letter that must be generated by the sender $\mathcal{B}$ in order to create the message. $\tau_2 + \mathcal{D}_2 - \tau_{\mathcal{A}} \leq \mathcal{L}_f \wedge \tau_2 + \mathcal{D}_2 - \tau_{\mathcal{B}} \leq \mathcal{L}_f$ is a conjunction of two simple temporal conditions that should be fulfilled to enable the execution of this step. This conditions are similar to the previous one but take into account the time τ_2 respectively.

$$\begin{aligned} &\alpha_3 = (\alpha_3^1, \alpha_3^2), \\ &\alpha_3^1 = (\mathcal{A}; \mathcal{B}; \langle \tau_{\mathcal{B}} \rangle_{\mathcal{K}_{\mathcal{B}}}), \\ &\alpha_3^2 = (\tau_3; \mathcal{D}_3; \{\tau_{\mathcal{A}}, \mathcal{K}_{\mathcal{B}}\}; \{\emptyset\}; \tau_3 + \mathcal{D}_3 - \tau_{\mathcal{A}} \leq \mathcal{L}_f \wedge \tau_3 + \mathcal{D}_3 - \tau_{\mathcal{B}} \leq \mathcal{L}_f). \end{aligned} \tag{5}$$

The third step should be considered analogously.

In our approach this formally defined protocols are interpreted in specially constructed mathematical computational structure that models protocols' executions in real computer networks. A proper definition can be found in [9].

In this structure we consider a finite set of network's users, a finite set of cryptographic primitives (keys, timestamps, etc.). Using this and a formal definition of a protocol we can combinatorically determine a set of hypothetical protocol's execution using a simple substitusions from the formal descripions to the sets in a structure. These executions are hyphotetical in that sense that sometimes they cannot be executed in real networks because of the lack of user's knowledge.

Let us consider the following example: we can determine a hyphotetical execution of the NSPKL protocol by substitution for a sender $\mathcal{A}$ a receiver by the Intruder $\mathcal{I}$ that impersonates the user $\mathcal{B}$ (we describe this situation by $\mathcal{I}(\mathcal{B})$). We also assume that $\mathcal{A}$ and $\mathcal{I}$ use their own timestamps $\tau_{\mathcal{A}}$ and $\tau_{\mathcal{I}}$. Thus, we have the following execution:

$$\begin{aligned}
&\alpha_1 \;\; \mathcal{A} \rightarrow \mathcal{I}(\mathcal{B}) : \langle \tau_{\mathcal{A}} \cdot \mathcal{I}_{\mathcal{A}} \rangle_{\mathcal{K}_{\mathcal{B}}}, \\
&\alpha_2 \;\; \mathcal{I}(\mathcal{B}) \rightarrow \mathcal{A} : \langle \tau_{\mathcal{A}} \cdot \tau_{\mathcal{I}} \cdot \mathcal{I}_B \rangle_{\mathcal{K}_{\mathcal{A}}}, \\
&\alpha_3 \;\; \mathcal{A} \rightarrow \mathcal{I}(\mathcal{B}) : \langle \tau_{\mathcal{B}} \rangle_{\mathcal{K}_{\mathcal{B}}}.
\end{aligned}$$

Observe that the Intruder cannot decrypt the first message because it doesn't possess the private key of the user $\mathcal{B}$ and this hypothetical execution cannot be executed in the real network. However remember that it can be executed as a part of interleaving the attacking execution, which was discovered and presented by Lowe in [10]. Using boolean encoding and SAT solvers we can determine whether hypothetical executions can be executed or not. This procedure was described in our previous works, for example in [9].

Here we omit a detailed presentation of the computational structure because of the lack of space.

Now we will introduce several types of time parameters that characterize some time aspects during and after the protocol's execution. In the next formulas we will use the following notations:

- k is a step number,
- i is a step counter $i = k + 1...n$,
- n is a number of protocol's steps,
- T_k^{out} is a lifetime in the k-th step,
- T_i^{max} is the maximum step time.

In the computational structure we define the following time dependencies.

Firstly, we determine that the current step time T_k is a sum of: encryption time T_e, generation time T_g, current delay in the network D, and decryption time T_d:

$$T_k = T_e + T_g + D + T_d. \tag{6}$$

The minimum step time (T_k^{min}) is a sum of: encryption time (T_e), generation time (T_g), minimum delay in the network (D_{min}) and decryption time (T_d):

$$T_k^{min} = T_e + T_g + D_{min} + T_d. \tag{7}$$

The maximum step time (T_k^{max}) is a sum of: encryption time (T_e), generation time (T_g), maximum delay in the network (D_{max}) and decryption time (T_d):

$$T_k^{max} = T_e + T_g + D_{max} + T_d. \tag{8}$$

Current session time (T_{ses}) is a sum of all current step times:

$$T_{ses} = \sum_{k=1}^{n} T_k. \tag{9}$$

The minimum session time (T_{ses}^{min}) is a sum of all minimum step times:

$$T_{ses}^{min} = \sum_{k=1}^{n} T_k^{min}. \tag{10}$$

The maximum session time (T_{ses}^{max}) is a sum of all maximum step times:

$$T_{ses}^{max} = \sum_{k=1}^{n} T_k^{max}. \tag{11}$$

Finally, we state that the lifetime should be a sum of all maximum session times in all protocol's steps:

$$T_k^{out} = \sum_{i=k}^{n} T_i^{max}. \tag{12}$$

From our current research point of view, the most important protocols time parameter is a lifetime. The lifetime defines a maximal waiting time for response during the execution of protocols steps.

3 Experimental Results

Our research on security protocols has been carried out in three stages. In the first stage, using the SAT-solver MiniSAT we verified the fact of the real execution possibility of generated combinatorically protocol's hyphotetical executions. The second stage of the research concerned the analysis of the duration of the protocols executions. In the last stage, simulations of the protocol's executions were made. In this stage values of delays in the network were randomly generated according to the following probability distributions:

- uniform distribution,
- normal distribution,
- Poisson's distribution,
- Cauchy's distribution,
- exponential distribution.

The selected probability distributions simulate the operation of a computer's networks. Uniform and normal distributions reflect the real work of the network. Poisson's and Cauchy's distributions show a heavy loaded network. Exponential distributions simulate a fast network.

During the analysis, various time aspects were considered. The time parameters values are described in more detail in the next Section. All tests were carried out using an abstract time unit [tu]. This unit is an arbitrarily selected and considered time period.

As we mentioned in the Introduction, a timed version of the NSPKL protocol was used for our experiments. It is only possible to perform the *man in the middle* attack on this protocol. The research involved an analysis of the impact of time parameters on the possibility of an attack by the Intruder.

Table 1. Summary of the NSPKL protocol executions

No.	Parts	Parameters	No.	Parts	Parameters
1	$A \rightarrow B$		10	$B \rightarrow A$	
2	$I \rightarrow B$	T_I, K_I	11	$I \rightarrow A$	T_I, K_I
3	$I \rightarrow B$	T_A, K_I	12	$I \rightarrow A$	T_B, K_I
4	$I(A) \rightarrow B$	T_I, K_A	13	$I(B) \rightarrow A$	T_I, K_B
5	$I(A) \rightarrow B$	T_A, K_A	14	$I(B) \rightarrow A$	T_B, K_B
6	$A \rightarrow I$	T_I, K_I	15	$B \rightarrow I$	T_I, K_I
7	$A \rightarrow I$	T_B, K_I	16	$B \rightarrow I$	T_A, K_I
8	$A \rightarrow I(B)$	T_I, K_B	17	$B \rightarrow I(A)$	T_I, K_A
9	$A \rightarrow I(B)$	T_B, K_B	18	$B \rightarrow I(A)$	T_A, K_A

Table 1 presents possibilities of the communication between users A and B, and vice versa. Column *No.* contains the execution's order number. This designation was introduced to simplify the way of referring to executions during their analysis. Column *Parts* contains information about the sender and the recipient in the first execution's step. The A and B indicate honest users, I indicates an Intruder, A as well as B indicate an Intruder impersonating honest users. In the column marked as *Parameters* elements which are used by the Intruder are located.

The first phase of NSPKL testing was related to determining the impact of the encryption time on the attack susceptibility. In this case, it was assumed that the delay in the network will take values from 1 to 3 [tu]. For the research, the value of the current delay in the network was equal to 1 [tu], encryption time changed by 1 [tu] from 1 to 10 [tu]. The tested attacking executions were executions no. 9 and 18.

A summary of the results obtained for both executions was collected in Table 2. This table contains information about the value of the encryption time, the calculated lifetime, and the current minimum and maximum session time. In addition, new lifetime values have been placed in brackets in column *Lifetime*. Those lifetime values should prevent the attacker from carrying out an attack.

For attacking executions it turned out that regardless of the encryption and decryption time value, the Intruder is able to carry out this attack. During that

Table 2. Summary of results for the NSPKL protocol

Encryption time [tu]	Lifetime [tu]			Time [tu]		
	Step 1	Step 2	Step 3	T_{ses}	T_{ses}^{min}	T_{ses}^{max}
1	17	11 (6)	5	14	11	17
2	23	15 (9)	7	20	17	23
3	29	19 (12)	9	26	23	29
4	35	23 (15)	11	32	29	35
5	41	27 (18)	13	38	35	41
6	47	31 (21)	15	44	41	47
7	53	35 (24)	17	50	47	50
8	59	39 (27)	19	56	53	59
9	65	43 (30)	21	62	59	65
10	71	47 (33)	23	68	65	71

executions the Intruder does not generate nor encrypt or decrypt the messages. He or she comes into possession of entire ciphertexts and sends them away. Thus, the time it takes to execute steps is always less than the minimum step time. The minimum time needed by the Intruder to attack was 14 [tu]. Next, the tool analyzed the NSPKL protocol again and designated new safe lifetime values and the minimum and maximum session time for the protocol.

As it has already been mentioned, the values of corrected lifetimes are included in brackets in Table 2. The lifetime values were modified in the second step because in order to execute this step, the Intruder had to execute additional steps to acquire the relevant knowledge. Thanks to the modified lifetime values in the second step, we can protect the network from an attack by the Intruder.

In each test series the session time was 3 [tu] greater than T_{ses}^{min}. With this result an interesting relationship is found. During this attack, the Intruder must execute three additional steps to acquire relevant knowledge. For the execution No. 9, additional steps were invoked from the execution 5. In Table 3 a summary of times of particular steps is presented.

Table 3. Summary of the attacking execution

No.	Basic step	Additional step	T_e	T_g	D_{min}	T_d
1	9.1		+	+	+	−
2		5.1	−	−	+	+
3		5.2	+	+	+	−
4	9.2		−	−	+	+
5	9.3		+	+	+	−
6		5.3	−	−	+	+

In the first step (9.1), an honest user generated his timestamp and encrypted the message. The Intruder impersonating another user was not able to decrypt this message, so the decryption time was not added. In the second step (5.1), the Intruder sent a ready cipher, so the encryption and generation times were not added. However, the time of delay and decryption (in the case of the honest user) were added to the value of the second step time. A similar thing happened in the next steps of the entire execution. This summary shows what times are included in the operations performed by users during the steps.

The encryption and decryption time took the same value. Thus, the following relationship can be determined:

$$T_{ses} = 6 \cdot T_e + 2 \cdot T_g + 6 \cdot D_{min}. \tag{13}$$

Then, due to the assumed values of generation times and delay in the network, the above dependence will take the form:

$$T_{ses} = 6 \cdot T_e + 8. \tag{14}$$

The correctness of this relationship has been confirmed analytically. The same values of the session times were obtained by the implemented tool.

The next stage of the research consisted in checking the network delays' impact on the attack's possibility. During the tests it was assumed that the encryption and decryption times were equal to 2 [tu] and the generation time was equal to 1 [tu]. The delay in the network's ranges changed during each test session from 1–4 [tu] to 1–10 [tu]. Of course, for the lifetime calculations the upper limit of the tested range was taken into account and the lower range limit (1 [tu]) was used to calculate the session times.

Attacking executions (9 and 18) were also tested. For all seven test series, the minimum time that the Intruder needed to execute the attack was 20 [tu]. The maximum session time differed in test series due to the delay in the network's range. Experiments have shown that a *man in the middle* attack was always possible, regardless of the upper limit of delay in the network.

The *man in the middle* attack on the NSPKL protocol proved to be possible regardless of the set values of the encryption times and delay in the network's ranges. To protect the NSPKL protocol from such attacks, lifetimes' replacement is necessarily, especially in the steps in which the Intruder is the sender. In these steps, the Intruder must acquire the knowledge. Limiting the time capability will allow securing the protocol against a *man in the middle* attack.

Next, the NSPKL protocol simulations were carried out. First, we used SAT-solver to check which executions are not possible to carry out. Their numbers are: 4, 5, 8, 13, 14 and 17. These executions were not included in the tests. After that, we assumed:

- $T_e = T_d = 2[tu]$,
- $T_g = 1[tu]$,
- delay in the network's range: $1 - 10[tu]$.

Next for those assumptions lifetimes were calculated ($T^1_{out} = 44[tu]$, $T^2_{out} = 29[tu]$, $T^1_{out} = 14[tu]$). Current delay values were randomly generated according to selected probability distributions: uniform, normal, Poisson's, Cauchy's, and exponential. Furthermore, we assumed four different endings of the execution. The first situation is *correct*. This means that the execution was completed in the correct time and time conditions have been preserved. The second situation is *!min*. It is for those executions which were completed with preserved time conditions, but the current session time was lower than T^{min}_{ses}. The next situation is *!max*. It is for those executions which were completed with preserved time conditions, but the current session time was greater than T^{max}_{ses}. The last situation is *error*. It is for those executions where any time condition was not fulfilled. The obtained results were presented on the example of a normal probability distribution.

Table 4. Experimental results for the NSPKL protocol and normal distribution

Execution no.	Correct	Lower then T^{min}_{ses}	Upper then T^{max}_{ses}	Error
1	231	0	242	527
2	216	0	144	640
3	0	0	107	893
6	224	0	161	615
7	0	0	23	977
9	0	0	0	1000
10	236	0	226	538
11	211	0	168	621
12	0	0	127	873
15	221	0	178	601
16	0	0	21	979
18	24	0	566	410

In Table 4 the summary of experimental results for the NSPKL protocol and normal distribution are presented. All the sessions of theb execution no. 9 ended with an error. The second attacking execution (18) ended correctly twenty-four times. Other sessions ended above T^{max}_{ses} (566) and with error (410). For the remaining executions, the correct, upper than T^{max}_{ses}, and error results appeared. No executions ended with a time lower than T^{min}_{ses}. The generated delays in the network's values caused more wrong sessions. For example, for execution no. 3 there were 107 test series which ended above T^{max}_{ses}, while 893 series ended with an error. These results reflect the natural work of a computer network. In such a network, messages flow in correct time intervals, but there are also problem situations.

In Table 5 the minimum, average and maximum values of the session time and average values of delay in the network were contained. Those series ended in the

Table 5. Timed values for the NSPKL protocol (series ended in correct time)

Execution no.	$Min(T_{ses})[tu]$	$Avg(T_{ses})[tu]$	$Max(T_{ses})[tu]$	$Ang(D)[tu]$
1	22.44	36.63	43.95	7.64
2	21.35	35.85	43.97	7.62
6	20.37	35.65	43.99	7.55
10	21.4	36.91	43.99	7.77
11	20.62	35.75	43.7	7.58
15	18.73	35.69	43.93	7.56
18	22.46	38.85	43.99	4.81

correct time. The minimum and maximum session times for specific encryption times enable observation of the Intruder's actions. If the delay in the network is relatively low, the session will be within the correct time interval. The Intruder will be able to carry out the attack.

Table 6 shows the minimum, average and maximum values of the session time and average values of delay in the network for series which ended above T_{ses}^{max}. The generated delay in the network's values affected a large number of error sessions. The presented results show that if the delay in the network is relatively high, the session time will exceed the correct time interval. Thus, the Intruder will not carry out the attack.

Table 6. Timed values for the NSPKL protocol (series ended aboveT_{ses}^{max})

Execution no.	$Min(T_{ses})[tu]$	$Avg(T_{ses})[tu]$	$Max(T_{ses})[tu]$	$Ang(D)[tu]$
1	45.32	52.14	73.28	12.86
2	44.05	50.95	72.87	12.65
3	51.48	76.04	97.37	8.83
6	44.01	52.35	72.41	13.12
7	50.55	72.25	102.75	8.53
10	44.12	52.47	71.7	12.99
11	44.02	51.13	74.49	12.71
12	48.36	75.54	100.49	8.9
15	44.01	51.37	78.86	12.79
16	45.25	64.32	89.28	7.27
18	44.19	83.07	115.69	11.42

4 Conclusion

The article presents a method for the analysis and verification of security protocols taking into account the time aspect. It has been shown that the study of time during individual steps of the protocol allows you to observe suspicious behaviors in the network: messages may come too soon or too late. To get this information we have to estimate the acceptable duration for steps of the session, which allows our model. The presented method of the analysis also allows us to observe a *man in the middle* attack.

The simulation results also affected the capabilities and activities of the Intruder. Different delay values mean that the Intruder sometimes had sufficient time to carry out an attack. In addition, random delays in the network have affected the communication of honest users. The high session time, which mimics problems that may occur in the real network, made the imposed time conditions unfulfilled and the communication ended with an error. During these steps, in which the delays in the network were equal to D_{max}, the communication of honest users was going well.

In further research, we can increase the number of parameters considered in a probabilistic manner (first works connected with it were shown in [13]), and also reject the assumption of the ideal cryptography. With this analysis, we can discover the strengths and weaknesses of the protocol and adapt it dynamically to the existing situation in the network.

References

1. Armando, A., et al.: The AVISPA tool for the automated validation of internet security protocols and applications. In: Etessami, K., Rajamani, S.K. (eds.) CAV 2005. LNCS, vol. 3576, pp. 281–285. Springer, Heidelberg (2005). https://doi.org/10.1007/11513988_27
2. Blanchet, B.: Modeling and verifying security protocols with the applied Pi calculus and proverif. found. Trends Priv. Secur. **1**(1–2), 1–135 (2016)
3. Burrows, M., Abadi, M., Needham, R.: A logic of authentication. ACM Trans. Comput. Syst. **8**(1), 18–36 (1990). https://doi.org/10.1145/77648.77649
4. Cremers, C.J.F.: The scyther tool: verification, falsification, and analysis of security protocols. In: Gupta, A., Malik, S. (eds.) CAV 2008. LNCS, vol. 5123, pp. 414–418. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-70545-1_38
5. Dolev, D., Yao, A.C.: On the security of public key protocols. Technical report, Stanford, CA, USA (1981)
6. Jakubowska, G., Penczek, W.: Is your security protocol on time ? In: Arbab, F., Sirjani, M. (eds.) FSEN 2007. LNCS, vol. 4767, pp. 65–80. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75698-9_5
7. Jakubowska, G., Penczek, W.: Modelling and checking timed authentication of security protocols. Fundam. Inform. **79**(3–4), 363–378 (2007)
8. Kurkowski, M.: Formalne metody weryfikacji własności protokołów zabezpieczających w sieciach komputerowych. Informatyka - Akademicka Oficyna Wydawnicza EXIT, Akademicka Oficyna Wydawnicza Exit (2013)

9. Kurkowski, M., Penczek, W.: Verifying security protocols modelled by networks of automata. Fundam. Inf. **79**(3–4), 453–471 (2007). http://dl.acm.org/citation.cfm?id=1366071.1366086
10. Lowe, G.: Breaking and fixing the needham-schroeder public-key protocol using FDR. In: Margaria, T., Steffen, B. (eds.) TACAS 1996. LNCS, vol. 1055, pp. 147–166. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-61042-1_43
11. Needham, R.M., Schroeder, M.D.: Using encryption for authentication in large networks of computers. Commun. ACM **21**(12), 993–999 (1978)
12. Paulson, L.C.: Inductive analysis of the internet protocol TLS. ACM Trans. Inf. Syst. Secur. **2**(3), 332–351 (1999)
13. Siedlecka-Lamch, O., Kurkowski, M., Piatkowski, J.: Probabilistic model checking of security protocols without perfect cryptography assumption. In: Gaj, P., Kwiecień, A., Stera, P. (eds.) Computer Networks, pp. 107–117. Springer International Publishing, Cham (2016)
14. Steingartner, W., Novitzka, V.: Coalgebras for modelling observable behaviour of programs. J. Appl. Math. Comput. Mech. **16**(2), 145–157 (2017)
15. Szymoniak, S., Siedlecka-Lamch, O., Kurkowski, M.: Timed analysis of security protocols. In: Grzech, A., Świątek, J., Wilimowska, Z., Borzemski, L. (eds.) Information Systems Architecture and Technology: Proceedings of 37th International Conference on Information Systems Architecture and Technology – ISAT 2016 – Part II. AISC, vol. 522, pp. 53–63. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-46586-9_5

A Note on Keys and Keystreams of Chacha20 for Multi-key Channels

Adam Czubak(✉), Andrzej Jasiński, and Marcin Szymanek

Institute of Mathematics and Informatics, Opole University,
ul. Oleska 48, 45-052 Opole, Poland
{adam.czubak,ajasi,mszymanek}@math.uni.opole.pl

Abstract. In this paper we analyze the keystreams generated by the Chacha20 stream cipher. We also compare these to the ones generated by its predecessor, the RC4 stream cipher. Due to the proposed multi-key channels in the upcoming TLS 1.3 standard we analyze the behavior of the keystream in the boundary case where there is a single bit difference between two keys used for the initiation of the stream cipher algorithms. The goal is to check whether a single bit change in the key has any predictable influence on the bits of the keystream output.

Keywords: Cybersecurity · Stream cipher · Symmetric encryption Chacha20

1 Introduction

Stream ciphers are nowadays used in many applications as one part of security solutions. Just to name a few:

- in SSL/TLS protocols for encrypting HTTP, FTP, SMTP, NNTP, XMPP traffic;
- in IPsec protocol suite for encrypting VPN tunneled packets between remote network sites;
- for encrypting mobile wireless communication channels (e.g. LTE, 5G); and many more.

The main advantage of symmetric encryption, and as such of stream ciphers, is the speed of the symmetric encryption process [1]. Symmetric, either stream or block ciphers, are used for actual encryption of large amounts of data in cryptographic solutions, whether the data is a document file, a flow of network packets or a raw stream of bits. For that reason symmetric ciphers are also referred to as bulk ciphers [2].

In the SSL and TLS protocols up to version 1.2 a single symmetric encryption key was used for a single connection[1]. The idea to change the symmetric key

[1] In the case of selected symmetric ciphers an additional initialization variable, the IV (i.e. initialization vector, so called nonce) is used. The IV is one of the initialization parameters for the symmetric cipher. Examples of ciphers employing an IV are AES and Chacha20.

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 357–372, 2018.
https://doi.org/10.1007/978-3-319-92459-5_29

during a single connection is introduced in the draft for TLS 1.3 [3,4] with the notion of a single master key, that is used to deterministically derive consecutive phase keys actually used to encrypt data during a single TLS connection. So the first master key is created mutually by both peers whereas the consecutive ones require local computation only. The concept is called multi-key channels and is described in Sect. 3.

In this paper we focus on the Chacha20 stream cipher that was introduced in 2008 as one of the finalists of the eStream Project [5,6]. Chacha20 was also chosen by Google LLC as one of preferred stream ciphers as a replacement for RC4 in TLS connections for its server cipher suites [7], namely:

- `TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256` and
- `TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256`.

We analyze the behavior of the keystream generated by Chacha20 in the edge case where there is a single bit difference between two keys used for the initiation of the stream cipher algorithm. The goal is to check whether a single bit change in the key has any predictable influence on the bits of the keystream output. As a reference to the results of Chacha20 analysis we chose the RC4 stream cipher, which is known to be vulnerable and has biased bits in its output keystream [8–11].

This paper is of TRL level of 3[2].

2 Preliminaries

2.1 Symmetric Ciphers

Symmetric-key encryption [1] provides secrecy when two parties, say Alice and Bob, communicate with each other. An adversary who intercepts a message should not get any significant information about its content. To set up a secure communication channel, Alice and Bob first agree on a key k. They keep their shared secret key k. Before sending a message (plaintext) m to Bob, Alice encrypts m by using the encryption algorithm E and the key k. She obtains the ciphertext $c = E(k, m)$ and sends c to Bob. By using the decryption algorithm D and the same key k, Bob decrypts c to recover the plaintext $m = D(k, c)$. It is required that the encrypted plaintext m can be uniquely recovered from the ciphertext c. This means that for a fixed key k, the encryption map must be bijective. Mathematically, symmetric encryption may be considered as follows.

A symmetric-key encryption scheme consists of a map

$$E : K \times M \mapsto C \tag{1}$$

such that for each $k \in K$, the map

$$E_k : M \mapsto C; m \mapsto E(k; m) \tag{2}$$

[2] TRL 3 (Technology Readiness Level: experimental studies to physically validate the analytical predictions for the technology.

is invertible. The elements $m \in M$ are the plaintexts (also called messages). C is the set of ciphertexts or cryptograms, the elements $k \in K$ are the keys. E_k is called the encryption function with respect to the key k. The inverse function $D_k = E_k^{-1}$ is called the decryption function. It is assumed that efficient algorithms to compute E_k and D_k exist.

2.2 Stream Ciphers

We consider a stream cipher which operates on a stream of plaintext bytes. Processing character by character, byte by byte, it encrypts plaintext strings of arbitrary size. Let us define a stream cipher formally. Let K be a set of keys and M be a set of plaintexts. In this context, the elements of M and K are called characters. A stream cipher

$$E^* : K^* \times M^* \mapsto C^*;\ E^*(k, m) = c = c_1 c_2 c_3 \cdots \tag{3}$$

encrypts a stream $m = m_1 m_2 m_3 \cdots \in M^*$ of plaintext characters $m_i \in M$ as a stream $c = c_1 c_2 c_3 \cdots \in C^*$ of ciphertext characters $c_i \in C$ by using a key stream $k = k_1 k_2 k_3 \cdots \in K^*$ of characters $k_i \in K$. The plaintext stream $m = m_1 m_2 m_3 \cdots$ is encrypted character by character. For this purpose, there is an encryption map E which encrypts the single plaintext character m_i with the corresponding key character k_i:

$$E : K \times M \mapsto C \qquad c_i = E_{k_i}(m_i) = E(k_i, m_i);\ i = 1, 2, \cdots \tag{4}$$

Encrypting plaintext characters with E_{k_i} is a bijective map for every key byte $k_i \in K$. Decrypting a ciphertext bytestream $c = c_1 c_2 c_3 \cdots$ is done character by character by applying the decryption map D with the same key stream $k = k_1 k_2 k_3 \cdots$ that was used for encryption:

$$c = c_1 c_2 c_3 \cdots \mapsto D(k, c) = D_{k_1}(c_1) D_{k_2}(c_2) D_{k_3}(c_3) \cdots . \tag{5}$$

Typically in practice, the characters in M and C and the key elements in K are binary digits, simply bytes. The encryption is then defined by $c = k \oplus m$, and decryption becomes $m = k \oplus c$, where $a \oplus b = a + b \mod 2$ (so called modulo 2 addition) for arbitrary binary strings $a, b \in \{0, 1\}^n$.

For the experiments discussed in Sect. 6 we will be utilising the difference between two bit sequences, namely the Hamming distance. The Hamming distance H_d between two strings of equal length is the number of positions at which the corresponding symbols are different [12].
In other words, it measures the minimum number of substitutions of bits required to change one string into the other. It's easy to observe that for binary strings a and b the Hamming distance is equal to the number of bits set to one in $a \oplus b$.

3 Multi-key Channel in TLS 1.3

Establishing secure multi-key channel implies that a sender and a receiver can update their keys during their communication [4], in contrast to the classical

situation where a single key is being used throughout the connection lifetime (Fig. 1). In such a case Alice and Bob share a symmetric key which is used by Alice to send messages to Bob using a secure channel.

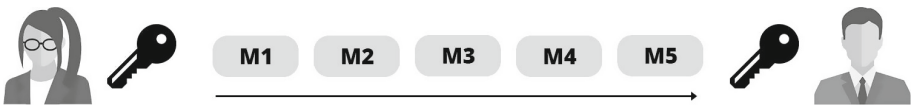


Fig. 1. Classical single symmetric key approach

Security of such channel originates from a single shared symmetric key and compromising such key compromises security of the entire channel. Key compromisation was and is a major issue taken under consideration in the development process of upcoming version 1.3 of TLS [3]. The development team considered the use of so called *multi-key channel* as a viable option to allow senders and receivers to update symmetric keys during the lifetime of the channel (Fig. 2). TLS 1.3 draft deploys a two-layer deterministic key schedule providing users with *master secret keys* and *phase keys* (as a key and IV update functionality). The first master secret key is derived from the key exchange (TLS handshake) whereas the consecutive master secret keys are derived deterministically from the prior one. TLS also implements the usage of phase keys to provide confidentiality and integrity in each phase. They are derived deterministically but in this case the input for their generation is the current master secret key. The phase keys are the ones actually used to encrypt the channel data.

Based on such premise the process of key creation as shown below (Fig. 2) begins with derivation of the first master from TLS handshake. Next the phase key is created based on the first master key. Only the phase key is used to secure (i.e. encrypt) the communication channel. After a certain time period or a certain amount of messages sent communication enters a second phase. A new master key is created based on its predecessor and as in the previous phase a new phase key is derived from the new master key. The process of master and phase key creation is repeated until the communication is over. Such multiple key solution allows to define two advanced security notions. The first notion defines that some security should be provided even if a master secret key or a phase key are compromised. Such notion, also called forward security, concentrates on a situation of full compromisation of the channel or corruption of a master secret key (Fig. 3).

Nevertheless, if the adversary gains access to the phase key (since it has access to the ciphertext encrypted with the phase key) and is able to decrypt the messages intercepted and sent in the phase corresponding to the compromised key, the messages which have been sent or will be sent using a different phase key should not be considered as compromised (Fig. 4).

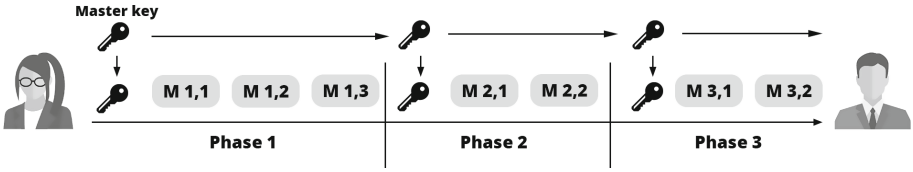


Fig. 2. Two-layer symmetric multi-key channel approach with master and phase keys

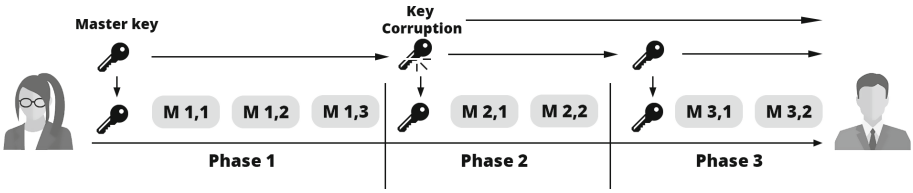


Fig. 3. Compromisation of a master key in a multi-key scenario

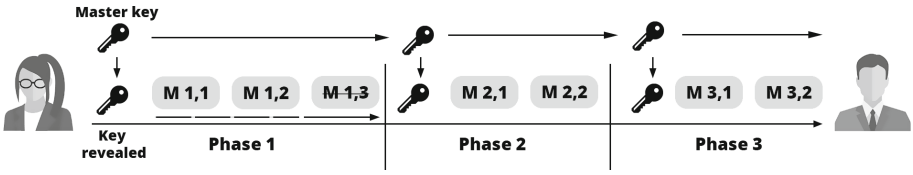


Fig. 4. Compromisation of a phase key in a multi-key scenario

4 Chacha20 Stream Cipher

Chacha cipher [5] is a modification of Salsa cipher [6], designed by D.J. Bernstein for eSTREAM project. ChaCha consist of a number of rounds - Chacha20 consist 20 rounds (10 double rounds), ChaCha8 and ChaCha12 respectively 8 and 12 rounds. Chacha is superior to Salsa in that it has better diffusion properties while retaining the same computational complexity as Salsa.

No fully successfull attack on ChaCha20 has been published in the literature but there are known attacks on ChaCha6 and ChaCha7. The most famous is the attack of Aumasson and others [13] which reduced the complexity of the successful attack on Chacha7 (with probability of 1/2) to 2^{248}.

The ChaCha20 is a stream cipher that operates on 64-byte data blocks. For each 64-byte data block, the algorithm calls the ChaCha20 expansion function. The input to the function is a secret key (which can have 32 or 16 bytes) and an 8-byte IV, additionally connected to the number of the currently encrypted block. The block number values change from 0 to $2^{64} - 1$ (so it is written on 8 bytes). Each invocation of the extension function increases the value of the block number by one. The core of the ChaCha20 algorithm is a mixing function that receives 64 bytes from the expansion function, then mixes them, and finally returns the other 64 bytes of data back to the expansion function. The mixing

function operates on data divided into 32-bit words arranged in a matrix of 4×4. The input matrix x may be represented as:

$$\begin{bmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{bmatrix}$$

The main role in ChaCha cipher is played by the function $QuarterRD$. It takes four words (a, b, c, d) as the input and gives four words as the output after performing following 32-bit operations:

$$\begin{aligned} a+ &= b;\ d\oplus = a;\ d <<<= 16; \\ c+ &= d;\ b\oplus = c;\ b <<<= 12; \\ a+ &= b;\ d\oplus = a;\ d <<<= 8; \\ c+ &= d;\ b\oplus = c;\ b <<<= 7; \end{aligned}$$

Inside the mixing function we call function $DoubleRd$, which takes matrix x as the input and returns matrix

$$z = DoubleRD(x) = DiagonalRD(ColumnRD(x)) \tag{6}$$

where:

– function $ColumnRD(x)$ calls the function $QuarterRD$ four times, one time for each column of matrix x, i.e.:

$$\begin{aligned} (y_0, y_4, y_8, y_{12}) &= QuarterRD(x_0, x_4, x_8, x_{12}); \\ (y_1, y_5, y_9, y_{13}) &= QuarterRD(x_1, x_5, x_9, x_{13}); \\ (y_2, y_6, y_{10}, y_{14}) &= QuarterRD(x_2, x_6, x_{10}, x_{14}); \\ (y_3, y_7, y_{11}, y_{15}) &= QuarterRD(x_3, x_7, x_{11}, x_{15}); \end{aligned}$$

– function $DiagonalRD(x)$ calls the function $QuarterRD$ four times, once for each diagonal of matrix y, i.e.:

$$\begin{aligned} (z_0, z_5, z_{10}, z_{15}) &= QuarterRD(y_0, y_5, y_{10}, y_{15}); \\ (z_1, z_6, z_{11}, z_{12}) &= QuarterRD(y_1, y_6, y_{11}, y_{12}); \\ (z_2, z_7, z_8, z_{13}) &= QuarterRD(y_2, y_7, y_8, y_{13}); \\ (z_3, z_4, z_9, z_{14}) &= QuarterRD(y_3, y_4, y_9, y_{14}); \end{aligned}$$

5 RC4 Stream Cipher

RC4 was developed by Ron Rivest from RSA Security in 1987. Officially, the name is an acronym for "Rivest Cipher 4".

The weakness of RC4 is that some byte sequences appear slightly more frequently in the RC4 encoding stream than others. In 2000, Fluhrer and McGrew [11] developed an attack based on this property. With this attack, you can extract the encryption stream from a gigabyte of random output data. In 2001,

Fluhrer et al. [9] discovered that among all possible RC4 keys, the statistical distribution of the first few bytes of the initial encryption stream is strongly non-random. By analyzing a large number of messages encrypted with the same key, one can obtain this key.

In February 2015, in the proposed standard RFC 7465 [8] in essence prohibited the use of a RC4 cipher in TLS connections. In 2005, Andreas Klein presented an analysis of the RC4 stream cipher showing more correlations between the RC4 keystream and the key [14].

RC4 algorithm is performed in a two-step scenario. In the first step an array S of 256 bytes is created. It is filled with bytes based on the value of the secret key before the encryption or decryption operation begins. To create the array, first in each cell of the table a number corresponding to the number of its position (counting from zero) is entered. Next an auxiliary variable j is created and initialized to 0. For each number i in the array (numbers take values from 0 to 255 in succession), two operations are performed:

1. the value of the auxiliary variable j is modified as follows:

$$j = (j + S[i] + key[i \mod keylength]) \mod 256, \tag{7}$$

   where key is a secret key and $keylength$ is the length of the secret key;
2. Then the number in the array in the current position i is exchanged with the number in the table in the position specified by the variable j.

In the second step keystream bytes are generated. They are produced on the basis of array S. Two auxiliary variables p_1 and p_2 are created and both are initialized to 0. Variables p_1 and p_2 are changed as follows:

$$p_1 = (p_1 + 1) \mod 256, \quad p_2 = (p_2 + S[p_1]) \mod 256. \tag{8}$$

Next we switch places of two elements in the array S at positions p_1 and p_2. Then a new auxiliary variable p_3 with value

$$p_3 = (S[p_1] + S[p_2]) \mod 256. \tag{9}$$

is created. Finally the value in the array S at position p_3 is the next byte of the key stream. If more bytes of the key stream are needed, the algorithm repeats all the above steps.

6 Experimental Analysis

6.1 The Plan of Experiments

Stream ciphers use a pseudorandom keystream for the actual encryption. A symmetric stream cipher contains a PRG (Pseudo Random Generator) that takes a key of finite size (32 bytes in the case of Chacha20) as an input and returns an infinite keystream of pseudorandom bytes.

The main property of the resulting keystream is that on the basis of previous keystream bits it should be impossible to calculate the probability of the value of the consecutive keystream-bit other then 0.5. Furthermore, it should not be possible to predict bits of the keystream by influencing bit values in the key.

In practice in the case of TLS 1.3, where the symmetric key is being altered during the connection (see Sect. 3), it would be advantages for an attacker if the change itself would allow him to calculate the probability of the next bit value different from 1/2.

So it is preferable for a stream cipher to possess the so called avalanche effect, where a small change in the input (the key) has a strong impact on the output (the keystream). If we were using a random number generator and not a pseudorandom one then by a one-bit change in the key the result on average would be that half of the output bits were flipped. But the impact has to be somewhat subtle, because if the two keystreams would differ more than a half (or significantly less then a half), then it would mean that the keystream is predictable. In the edge case where a slight difference of the key would cause all of the keystream bits to change, thus meaning that a certain bit in the key negates the keystream.

The goal of our experiments was to verify whether a single-bit change in the key causes an undesirable change in the resulting keystream. We hypothesize that the average number of bit-changes for every bit flipped in the keys is the same. So we wish to verify that whatever bit in the key is flipped, the resulting keystream differs on average by half from the keystream corresponding to the original unmodified key. We will analyse the variance (ANOVA) for a single factor [15] case in the experiments. Besides the ANOVA test, we will also calculate additionally the kurtosis and observe outliers of the distribution.

The general idea of the experiments was to generate a pseudorandom 32-byte key (and in the case of Chacha20, an IV is generated as well), initialize a stream cipher with this key and generate first 64-bytes of the keystream. We flip one bit in the key and generate a second 64-byte keystream. Then we compare the two generated keystreams by calculating the Hamming distance between the two. We flip every one of the 256 bits of the key to check whether a change in a certain bit makes the Chacha20 algorithm more or less eager to change the resulting keystream bits.

The size of the key was chosen to match the TLS requirements regarding stream ciphers. Chacha20 requires the key to be 32 bytes (256-bit key) long [16] whereas RC4 is more flexible, it allows for keys of the size between 5 and 256 bytes (40 to 2048-bit key). Additionally since TLS 1.2 is nowadays commonly used with stream cipher keys of sizes 128 or 256 bits the latter was chosen for both Chacha20 and RC4 experiments. It is worth mentioning that RC4 has never been used in neither SSL or TLS in the 256-bit key option. The strongest server cipher suite involving RC4 was the 128-bit option in the suite `TLS_RSA_WITH_RC4_128_SHA`. Nevertheless, we decided to initialize both stream ciphers with 256-bit keys to achieve a fair comparison between the two.

The size of 64 bytes for the keystream was chosen since this is the actual size of the Chacha20 state table (see Sect. 4). RC4 has the state table of the size 256 bytes, but it was shown that already the second output byte of RC4 keystream is biased towards zero [10].

For every bit flip in the key we perform $N = 1,000,000$ observations. Thus, we have 256 groups of N samples. The results are stored in an $x[i,j]$ matrix, where $i \in \{1,2,\ldots,256\}$ corresponds to the bit flipped in the key and $j \in \{1,2,\ldots,N\}$ is the observation id. The average Hamming distance $\bar{x}_i$ for the ith group in N observations and the average Hamming distance $\bar{x}$ for all observations are as follows

$$\bar{x}_i = \frac{1}{N}\sum_{j=1}^{N} x_{ij}\ , \qquad \bar{x} = \frac{1}{256 \cdot N}\sum_{x=1}^{256}\sum_{j=1}^{N} x_{ij}. \tag{10}$$

The variance $(s_i)^2$ for the ith group in N samples and the variance s^2 for all samples are:

$$(s_i)^2 = \frac{1}{N-1}\sum_{j=1}^{N}(x_{ij} - \bar{x}_i)\ , \qquad s^2 = \frac{1}{(256 \cdot N) - 1}\sum_{x=1}^{256}\sum_{j=1}^{N}(x_{ij} - \bar{x}). \tag{11}$$

6.2 Experimental Environment

The results presented in this Section were generated in the `Linux SMP 4.10.0-38` environment with a custom C program. Any cryptographic primitives used for experiments were from the `openssl 1.0.2g-1ubuntu4.10 amd64` Secure Sockets Layer toolkit - cryptographic utility except for the Chacha20 source code, which was made available by the author Daniel J. Bernstein[3] The experiments on the keystreams of RC4 and Chacha20 followed the logic below:

Listing 1.1. Chacha20 keystreams generation

```
#define N_EXPERIMENTS 1000000
for(int e=0; e< N_EXPERIMENTS; e++)
{
  //For every bit in the key of Chacha20
  for(int b=0;b<256;b++){
    //Key & KeyStream generation
    key = GenerateRandom32BKey();
    iv = GenerateRandom8BIV();
    keyStream1 = ChaCha20_64BKeyStreamFrom32BKey(key,iv);
    FlipOneBitFrom32BKey(key,b);
    keyStream2 = ChaCha20_64BKeyStreamFrom32BKey(key,iv);
    int h = HammingDistance(keyStream1,keyStream2);
    if (b!=255) fprintf(file, "%d,", h);
    else fprintf(file, "%d\r\n", h);
  }
}
```

[3] D. J. Bernstein maintains a website https://cr.yp.to with his original Chacha20 implementations ported to diferent platforms https://cr.yp.to/chacha.html.

The number of experiments performed on every bit was equal to 1,000,000 so the amount of data generated for RC4 and Chacha20 stream ciphers was (2 * 8.2) GB of key data and (2 * 16.4) GB of keystream data.

It is worth mentioning that contemporary acknowledged statistical test suites like FIPS 1401, FIPS 1402, ENT tests, Diehard battery and NIST Statistical Test Suite (a statistical test suite for testing the pseudorandom number generators used in cryptographic applications) require around 12 MB of data only.

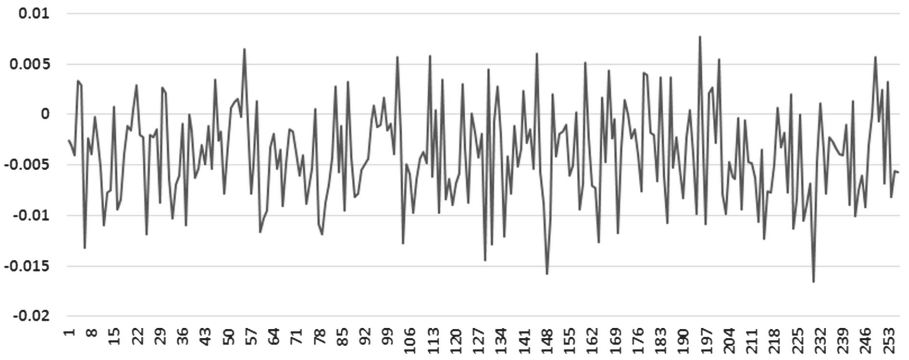


Fig. 5. Kurtosis for individual bits of Chacha20 key

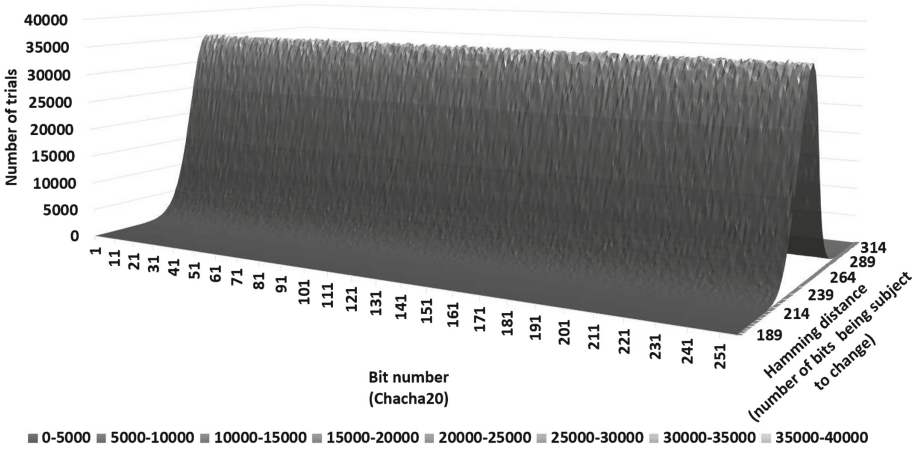


Fig. 6. Surface graph for Chacha20 key bits - 1st slope

6.3 Experimental Analysis of Chacha20 and RC4

Table 1 sets together experimental outcome of Hamming distance basic statistical measures. Results as shown above led us to believe that the Chacha20 and RC4 algorithms behave very decently in the matter of its affinity to Normal distribution as show in Figs. 6 and 7 for ChaCha20 and in Figs. 9 and 10 for RC4. The ANOVA tests results for RC4 and ChaCha20 are depicted in Table 2. Since in both cases $F < F_{test}$ then there are no grounds to reject the hypotheses that whatever bit in the key is flipped, the resulting keystream differs on average by half from the keystream corresponding to the original unmodified key.

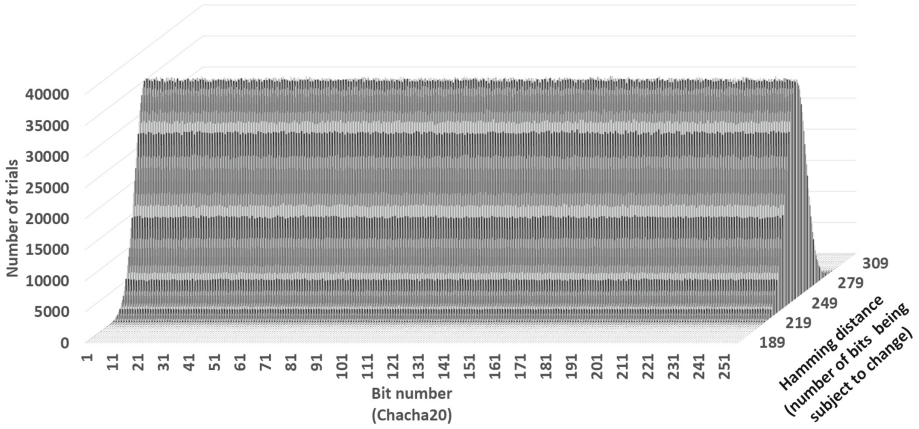


Fig. 7. Bar graph for Chacha20 key bits - 1st slope

Table 1. Hamming distance analysis summary

	Hamming distance		
	$\bar{x}$ for all samples	For individual bits	
		$\bar{x}_i$ range	Standard deviation s_i
ChaCha20	256.0012406	$<255.961468; 256.038078>$	$<11.28922932, 11.33932677>$
RC4	255.9953467	$<255.973198, 256.032373>$	$<11.29190835; 11.33345912>$

The graph of the kurtosis for individually changed bits is shown in Fig. 5 for ChaCha20 and Fig. 8 for RC4. It allowed us to see common irregularity in regard of the results affinity to the mean. The kurtosis analysis can be found in Table 3. The conclusions from Table 3 are that for the majority of bit changes (202 bits for ChaCha20 and 203 for RC4) in the data set we can observe a number of extreme results (located far away from the average). However, the value of kurtosis for ChaCha20 is higher, so the distribution is more concentrated than the RC4 distribution (see Figs. 5 and 8).

Table 2. ANOVA tests results for ChaCha20 and RC4

	F	F_{test}	α
ChaCha20	1.072083945	1.149991600	0.05
RC4	0.962842341	1.115042844	0.05

Table 3. Kurtosis analysis for ChaCha20 and RC4

	Bits changed	Kurtosis	Minimum value of kurtosis	Maximum value of kurtosis	Total value of kurtosis
ChaCha20	202	Negative	−0.016547312	0.007699836	−0.0038843
RC4	203	Negative	−0.017891875	0.007006858	−0.004012269

Therefore, we analyzed the concentration of results away from the average (see Table 4) interested in extreme results, i.e. upper and lower outliers, observations that adequately meet the conditions $H_d \leq Q1 - 1,5 \cdot IQR$ and $H_d \geq Q3 + 1,5 \cdot IQR$, where $Q1, Q3$ should be understood as the first and third quartile, whereas $IQR = Q3 - Q1$ is the interquartile range (see Table 4).

Table 4. Quartiles for ChaCha20 and RC4

	Quartile		IQR
	Q1	Q3	
ChaCha20	248	264	$264 - 248 = 16$
RC4	248	264	$264 - 248 = 16$

The graph in Fig. 11 shows the number of outliers, both the lower and upper for both RC4 and ChaCha20 for individual bits. One can notice quite a diversified distribution. For Chacha20 on average in 2655 samples there was no more than 224 bit change, and on average 2655 samples had no less than 288 changes. In the case of RC4 on average in 2663 samples there was no more than 224 bit change and on average 2650 samples had no less than 288 changes.

One could say that these results coincide. However there are changes in the bits for which the number is greater - i.e. in the case of ChaCha20 for bit 199 in 2801 samples the change has taken place in not less than 288 bits, while for bit 196 in 2797 samples there were changed no more than 224 bits whereas for RC4 for bit 60 in 2808 samples the change has taken place in not less than 288 bits, while for bit 183 in 2802 samples there were changed no more than 224 bits.

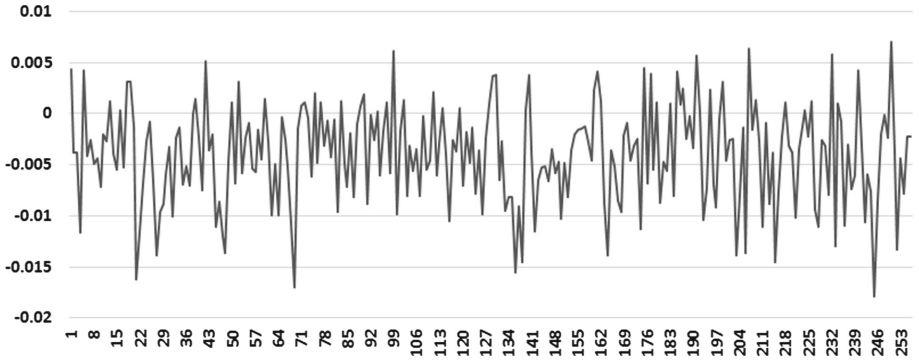


Fig. 8. Kurtosis for individual bits of RC4 key

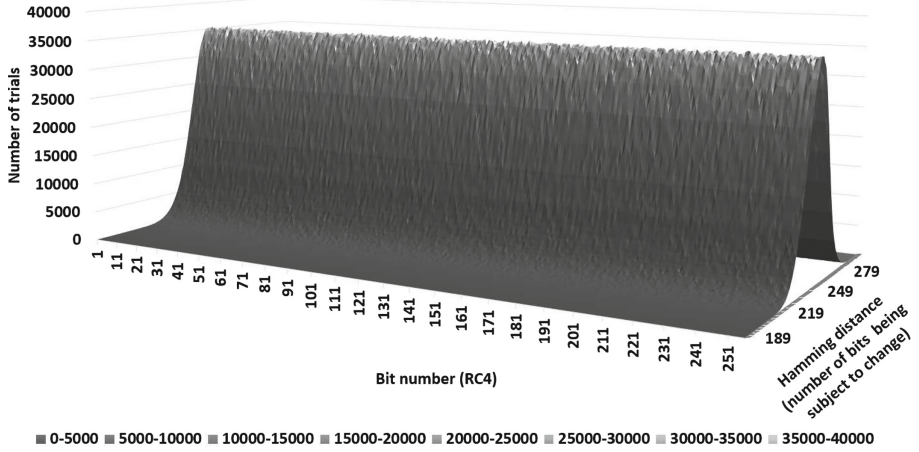


Fig. 9. Surface graph for RC4 key bits - 1st slope

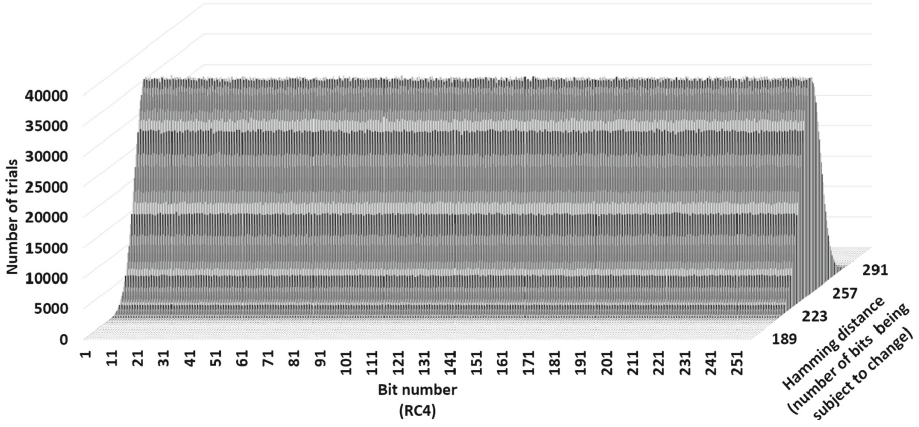


Fig. 10. Bar graph for RC4 key bits - 1st slope

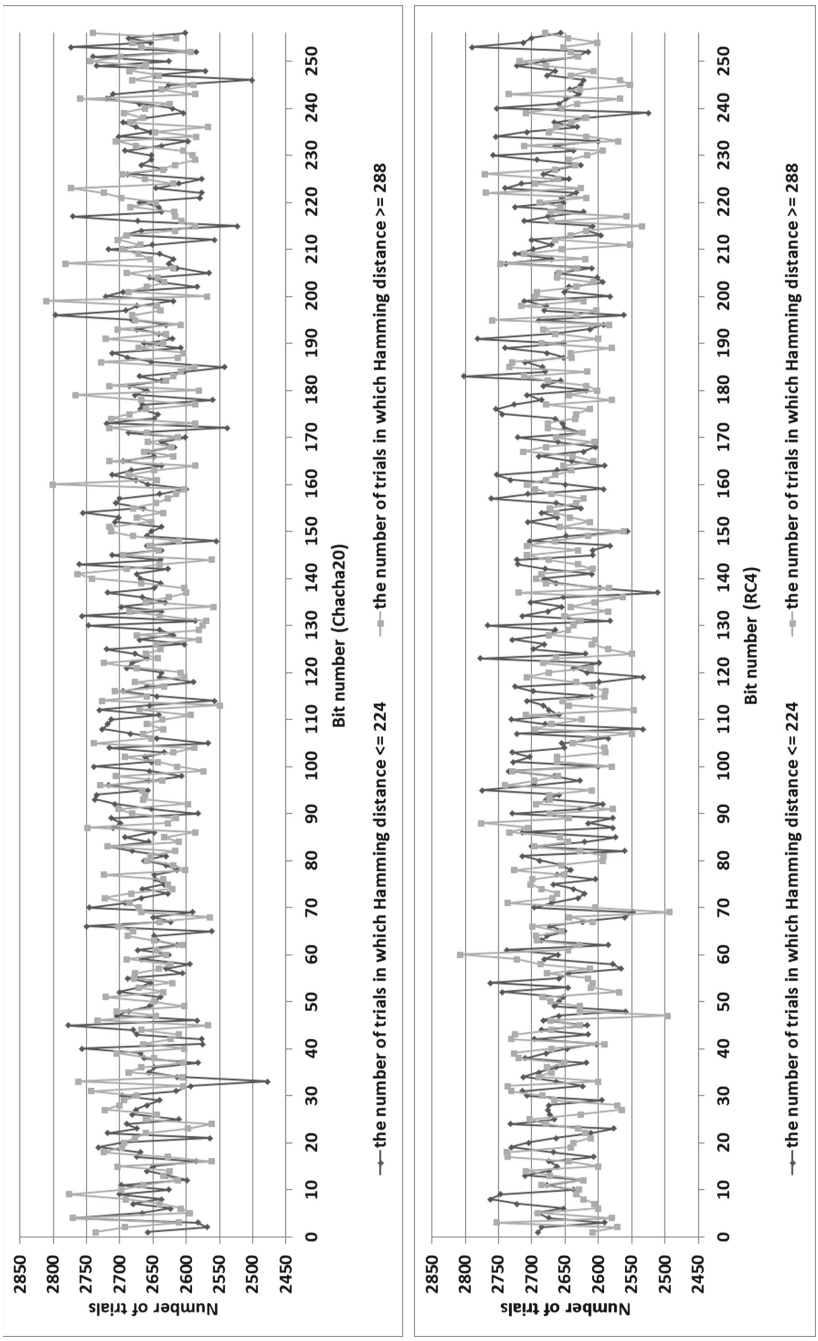


Fig. 11. Number of outliers and their per bit distribution for Chacha20 and RC4

7 Conclusions and Future Work

In this paper we have developed a set of key generators which allowed us to generate an observation database of changes occurring in the keystream of Chacha20 and RC4 encryption algorithms in a bit per bit manner. As a result we were able to conduct several statistical tests, data from which allowed us to analyze quantitative and qualitative characteristics of ChaCha20 and RC4 algorithms.

We investigated the boundary case where two keys differ in a single bit and compare the resulting keystreams. This is important for the TLS 1.3 protocol in which the idea of multi-key channels is introduced, where there are many symmetric keys used during a single connection.

As we were able to conclude, RC4 and ChaCha20 are very similar statistically in the manner of per bit change distribution, being both very close to the Gaussian distribution. Other statistical measures implied by such affinity to the normal distribution as quartiles, standard deviation and kurtosis have proven to be also extremely close to each other for both, RC4 and Chacha20 ciphers.

Such strong correlation suggests that further study should be conducted in order to determine if the similarities may impose a possible threat to ChaCha20, as is indicated by a noticeable skew of kurtosis in both cases.

In our opinion, the occurrence of extremely distant elements from the average in the distribution (outliers), suggests that there is some sort of a characteristic of Chacha20 that might be a potential vector of attack. For future work we wish to analyze in detail the nature of outliers found in the samples, as we believe that might lead to a discovery of a potential security threat. Additionally, we plan to analyze whether single-bit changes in the key tend to gather or are these evenly distributed throughout the keystream.

References

1. Paar, C., Pelzl, J.: Understanding Cryptography: A Textbook for Students and Practitioners. Springer, New York (2010). https://doi.org/10.1007/978-3-642-04101-3
2. Dierks, T., Rescorla, E.: The transport layer security (TLS) protocol version 1.2. In: Internet Requests for Comments RFC 5246 (2008). http://www.rfc-editor.org/rfc/rfc5246.txt
3. Rescorla, E.: The transport layer security (TLS) protocol version 1.3. In: IETF Network Working Group - Internet-Draft (2018). https://tools.ietf.org/html/draft-ietf-tls-tls13-23
4. Günther, F., Mazaheri, S.: A formal treatment of multi-key channels. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10403, pp. 587–618. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63697-9_20
5. Bernstein, D.J.: ChaCha, a variant of Salsa20. In: Workshop Record of SASC 2008: The State of the Art of Stream Ciphers. http://cr.yp.to/chacha/chacha-20080128.pdf
6. Bernstein, D.J.: The salsa20 family of stream ciphers. In: Robshaw, M., Billet, O. (eds.) New Stream Cipher Designs. LNCS, vol. 4986, pp. 84–97. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-68351-3_8

7. Google Swaps Out Crypto Ciphers in OpenSSL. In: Infosecurity Magazine (2014). https://www.infosecurity-magazine.com/news/google-swaps-out-crypto-ciphers-in-openssl/
8. Popov, A.: Prohibiting RC4 cipher suites. In: Internet Requests for Comments RFC 7465 (2015). http://www.rfc-editor.org/rfc/rfc7465.txt
9. Fluhrer, S., Mantin, I., Shamir, A.: Weaknesses in the key scheduling algorithm of RC4. In: Vaudenay, S., Youssef, A.M. (eds.) SAC 2001. LNCS, vol. 2259, pp. 1–24. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45537-X_1
10. Mantin, I., Shamir, A.: A practical attack on broadcast RC4. In: Matsui, M. (ed.) FSE 2001. LNCS, vol. 2355, pp. 152–164. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45473-X_13
11. Fluhrer, S.R., McGrew, D.A.: Statistical analysis of the alleged RC4 keystream generator. In: Goos, G., Hartmanis, J., van Leeuwen, J., Schneier, B. (eds.) FSE 2000. LNCS, vol. 1978, pp. 19–30. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44706-7_2
12. Hamming, R.W.: Error detecting and error correcting codes. Bell Syst. Tech. J. **29**(2), 147–160 (1950). https://doi.org/10.1002/j.1538-7305.1950.tb00463.x
13. Aumasson, J.-P., Fischer, S., Khazaei, S., Meier, W., Rechberger, C.: New features of latin dances: analysis of salsa, chacha, and rumba. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 470–488. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-71039-4_30
14. Klein, A.: Attacks on the RC4 stream cipher. In: Designs, Codes and Cryptography, vol. 48(3), pp. 269–286. Springer, Heidelberg (2008). https://doi.org/10.1007/s10623-008-9206-6
15. Brandt, S.: Data Analysis Statistical and Computational Methods for Scientists and Engineers. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-03762-2
16. Langleyand, A., Chang, W., Mavrogiannopoulos, N., Strombergson, J., Josefsson, S.: ChaCha20-Poly1305 cipher suites for transport layer security (TLS). In: Internet Requests for Comments RFC 7905 (2016). http://www.rfc-editor.org/rfc/rfc7905.txt

Security Considerations of Modern Embedded Devices and Networking Equipment

Błażej Adamczyk(✉)

Silesian University of Technology, Gliwice, Poland
blazej.adamczyk@polsl.pl
http://sploit.tech/

Abstract. The aim of this paper is to present the potential impact and risks related with security breaches in modern networking equipment and embedded devices in general. Firstly, the possible attack vectors and exemplary exploitation methods are presented. The methods are based on real vulnerabilities the author has recently found in a popular wireless router software. Besides presenting the vulnerabilities themselves the papers main goal is to assess the possible impact of a successful attack. Author presents several post exploitation methods which show how easily it is to use the fully-featured embedded device operating system maliciously.

Keywords: Embedded device security · Router exploitation
Network protection · MITM attack · Packet sniffing

1 Introduction

The spread of Internet of Things in recent years had a great influence on the hardware availability and prices. Thus, more and more frequently small embedded devices are equipped with powerful hardware configurations. This obviously is an advantage and can greatly increase the functionality of those devices. Taking a wireless router as an example of an embedded device, we can see this tendency - modern routers provide CPU and memory which is sufficient to run almost fully functional general purpose operating system. Quite often such a device is not only a simple router and firewall but additionally can be a VPN server or client, a network storage (by means of USB disk support), a web server, a proxy server, and so on.

Unfortunately the increase of complexity of an embedded system poses new security risks and threats. Firstly, the probability of a security bug is obviously greater in complex systems. Secondly, an attacker who gained access to such an embedded device can use a palette of features of a fully functional operating system to cause a greater harm.

In this paper the author presents a vulnerability discovered in a certain type of wireless routers and basing on this example tries to present security risks

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 373–384, 2018.
https://doi.org/10.1007/978-3-319-92459-5_30

related with a successful attack against such devices in general. It is important to understand that similar vulnerabilities have been recently discovered in many devices and majority of them gave the potential attacker the same wide range of possibilities because of a fully-functional operating system running on the device. In the author's opinion it is important to know what is at stake and what are potential consequences of a successful security breach in modern embedded devices.

The paper is structured as follows. The embedded device attack surface is characterized in Sect. 2. In Sect. 3 an exemplary attack on ASUS wireless routers using the vulnerabilities discovered by the author of this paper are presented. Further in Sect. 4 consequences of such a successful attack on modern embedded devices are discussed basing on the given example. Finally, the paper is summarized in Sect. 5.

2 Attack Surface

Embedded devices are usually tailored for certain statically defined use cases. The specific area of application frequently requires low level access to hardware components. This results in software written usually in native languages (most frequently it is C language) what is good while considering the device's limited resources. Unfortunately in the case of security the native programming languages and their compilers do not employ sufficient security measures. For example in C language there are several functions[1] available which do not force sufficient bounds checking thus allowing for the memory override which further can lead to Remote Command Execution (RCE).

Further, the embedded devices connect several hardware and software components together in a tightly coupled manner. These components are independent from one another (frequently being produced by different entities) and thus need to be integrated by the device vendor to make it work as a single system. To achieve this integration, they usually choose to integrate all the software pieces together via the common operating system. This further creates another type of an attack vector: when one executable needs to execute the another one with proper parameters, it usually spawns a system shell in order to do this. This potentially causes the possibility to inject additional system commands in the parameters if these are not properly escaped. Such attack also leads to the RCE.

The last but not least are logical errors. Copying buffers manually without bounds checking can cause the same effect as the aforementioned C functions in the first paragraph. Errors in the authorization and authentication logic may lead to privilege escalation, often resulting in full system access. Similarly, the session handling logic[2] is also prone to session hijacking attacks with similar consequences, an example is another vulnerability found by the author of this paper [1].

[1] Functions in C without bounds checking: sprintf, vsprintf, snprintf, vsnprintf, memcpy, mempcpy, memmove, memset, strcpy, stpcpy, strncpy, strcat, strncat.

[2] For example, session handling in a HTTP server.

Everything discussed so far regards to actual software bugs but the real question is whether and how the bugs can be exploited by hackers. A part of the system which could be "reached" by an attacker (i.e. the system external interface) is frequently called an attack surface. This is the space over which the attacker has at least a limited control thus has the opportunity to interact with the device software.

Of course the attack surface can cause execution of only limited execution paths of the software. In other words, the code coverage corresponding to the attack surface is usually small. This greatly limits possibilities of the attacker but does not guarantee security.

The attack surface of an embedded device depends on the application. Network attached devices usually expose their interfaces through network services and protocols, USB and serial devices allow for interaction through a dedicated cable and protocol and so on. Almost always, however, there is some software which is responsible for handling the interface input data. An error in the handling software may allow the attacker to select such an input which will cause the software to misbehave.

Typical networking device's attack surface corresponds to the networking protocols the device supports[3]. Thus, the software responsible for handling network protocols can be:

- a part of kernel the networking stack including device drivers and firmware (e.g. 802.2, 802.11, ARP, IP, TCP), or
- a separate user-space application handling application layer protocols (e.g. TLS, HTTP, UPnP).

In traditional servers and computers an error in kernel protocol implementation gives the attacker full system access (as the protocol is handled by the kernel itself). On the other hand, user-space processes are usually executed with dedicated user contexts which means the attacker only gains control and access rights which are assigned to the hacked software. In such a case, if the attacker wants to gain full system access (administrative rights), he needs to find another vulnerability to elevate his privileges.

Unfortunately in the case of embedded devices, author's observations show that the user-space processes are usually all executed with the same administrative user account thus allowing the attacker to cause the same full system damage regardless of which process has been compromised.

Worth mentioning is also the fact that kernel protocol implementations are usually a part of an existing, widely used and tested operating system what makes them more secure than purpose built, rarely used user-space applications. This is the reason why security researchers usually focus on user-space applications and services.

Not to be groundless, let us consider several practical examples of router and embedded device vulnerabilities:

[3] Attacker can remotely detect which services are supported by the device using tools like *nmap*.

CVE-2017-9417 "Broadpwn" [4]. An example of a kernel based vulnerability. It affects a proprietary (Broadcom) wireless adapter driver. The vulnerability affected great number of devices, including highly secured mobile phones. The bug allowed for RCE in kernel space by sending properly crafted IEEE 802.11 packets. What is more - the vulnerability was present in the wireless' adapter firmware which is being executed in a separate chip (WiFi chipset) having direct kernel memory access but external to the main device processing unit, thus classical operating system security features are unable to help here.

CVE-2016-10229 Linux UDP RCE [9]. Another kernel based vulnerability in the Linux networking stack responsible for handling the UDP protocol. The bug potentially allows one for RCE in the kernel space by sending specifically crafted UDP datagrams. A system is vulnerable if some process (e.g. an UDP listening application) calls the *recv* system call with MSG_PEEK flag.

CVE-2014-9583 ASUS Infosrv [11]. A vulnerability affecting ASUS wireless routers. The bug is in the Infosrv executable which is a custom service listening on port 9999 UDP and its purpose is to allow one for router detection and configuration. The service has an error in authentication logic where the MAC address is not properly verified for incoming requests. This allows remote attackers to bypass the authentication and further execute arbitrary commands. Because the Infosrv process runs with administrator privileges, the attacker gains full system permissions.

CVE-2017-6548 ASUS Networkmap [6]. Another ASUS routers vulnerability has been found recently in the Networkmap binary. Its purpose is to detect services running in LAN using the multicast SSDP protocol. The process sends a multicast SSDP discovery message and compatible LAN devices respond with IP addresses, ports their services are listening on.

Unfortunately the binary has several buffer overflows in the response parsing logic which allows one for taking control over the running process. Again, the process is executed with administrator privileges, thus the attacker gains full system permissions.

D-Link UPnP RCE [7]. Similar vulnerability to the previous one but this time in D-Link routers. Here a router can be detected by computers in LAN using the UPnP protocol which is based on the same SSDP multicast search mechanism as the previous one. The binary improperly handles the "ST:uuid:" header and its value is then further used to spawn a shell sub-process without being properly escaped. This allows one for injecting an arbitrary shell command. Again, the process runs with administrative privileges.

CVE-2015-7755 Juniper SSH backdoor [8]. Security researchers at Fox-IT had found an interesting vulnerability in Juniper SSH and Telnet binaries. In fact, this vulnerability can be classified as a backdoor because the attacker is able to authenticate by providing any username and a statically defined password: "*<<<%s(un= %s') = %u*".

TCP-32764 [15]. Another vulnerability has been affecting routers of several different vendors (Cisco, Linksys, and Netgear). An executable called scfgmgr is running on the devices and is listening on TCP port 32764. The service handles messages of a custom protocol and allows one to write and read router configuration without authentication. For example the message id 1 allows one to dump nvram contents - i.e. present all the router configuration and passwords. This information is sufficient to access the router and take full control.

Many more router and embedded device vulnerabilities have been found up till now. Reverse Shell Security team has even developed a framework gathering exploits and simplifying the process of exploitation of router vulnerabilities called routersploit [12]. At the time of writing this article it contained, among others, 109 exploits addressing vulnerabilities in different routers.

As one can see, the surface area for attacking embedded devices is vast. Beginning with the low level kernel drivers and networking stack code, through low level networking services and finishing at the complex services like HTTP or SSH.

3 New Vulnerability and Its Exploitation Method

In this Section a newly discovered vulnerability in an ASUS router's HTTP server will be presented. It was disclosed recently by the author of this paper and received the following CVE identifier: CVE-2017-15655 [2].

An unauthorized attacker having access to the router's HTTP server can exploit this vulnerability to gain remote RCE with full administrative rights. It affects all ASUS routers with firmware version up to and including 3.0.0.4.376. Moreover, despite the vulnerability is now fixed, some old ASUS routers which are not supported anymore are left running without the patch. Vendor has refused to publish a hot-fix to the unsupported routers. According to Shodan [13][4], at the time of writing this paper, there have been still more than 80 000 devices vulnerable worldwide which have the administration HTTP server exposed to the Internet.

The author has additionally tested the hosts reported by Shodan to make sure these are really the vulnerable devices by requesting an unauthenticated resource (CSS file) which is strictly related with the vulnerable ASUS firmware. Testing all the 82 786 hosts returned by Shodan gave the following results:

- 49 727 hosts (60%) returned the expected CSS resource - the vulnerable devices,
- 26 710 hosts (32%) were offline[5],
- 6 483 hosts (8%) did not return the CSS file - probably broken routers or honeypots.

[4] The Shodan's query used to find all vulnerable devices was: ""RT-" httpd Unauthorized" [14].

[5] Unfortunately these devices are usually home and small office routers and can frequently be offline. Shodan verifies them continuously - thus, it is very probable that a part of the offline devices can appear online in the near future.

Probably there exist much more devices running the vulnerable firmware which have the administration interface available only on LAN (this is the default setting) - thus were not indexed by Shodan. Such devices are also vulnerable but require access to LAN (e.g. access to the wireless network).

ASUS routers run a firmware containing the Linux kernel and a filesystem with supporting user-space compiled programs. Routers may have ARM, MIPS (big endian) or MIPSEL (little endian) architecture. The firmware called ASUSWRT has been published open source and are available at ASUS support pages [5].

The router's web administration interface is served by an executable called httpd (located in */usr/sbin/httpd*). The binary is compiled from sources located in *release/src/router/httpd/*. The binary firmware file is a binary which merges several files and a header. The files merged are the kernel and the "squashed" file system. Using *binwalk* and *dd* tools it is possible to extract them. Then using *unsquashfs* one may extract the contents of the router's filesystem. Author has used the filesystem to run the httpd server in an emulator. To do this he used the open source user-space emulator (*qemu*) and in order to switch the root filesystem the *chroot* command has been used. The emulator offers additionally the "-g" parameter which allows one to connect a multi-architecture GNU Debugger (*gdb*) and debug the process.

The author has set some breakpoints at the interesting C functions (the functions without boundary checking, mentioned earlier) and sent a HTTP request to the httpd port to see if the binary copies some of the input data without checking the bounds. This revealed several possible buffer overflow errors. Two of them were directly available and could be triggered in unauthorized contexts. Namely the URL and the *Host* header were copied into global variables (variables stored at heap). The variables have a constant size - *url* is 128 bytes long and *host_name* is 96 bytes long. The httpd, while parsing, reads the request line by line, stores the line content in a local variable of size 10000 bytes, called *line* (here the bounds are properly processed). Finally, it parses the line and sets appropriate variables. This process is presented in Listing 1.1. The *host_name* and *url* variables are set using the values parsed from the *line* buffer without necessary bounds checking. Thus, if the input value for those variables is larger than the declared size, the buffer will get overflowed and it would be possible to override some part of the heap.

Knowing we can overflow the variables, it is necessary to find out what can be overwritten. This can be done using *objdump* tool, as presented in Listing 1.2. As we can see, the *host_name* global variable is stored in heap *.bss* section right after the *url* variable. Thus, a much better candidate for the overflow is the *host_name* because it can overflow more bytes remaining in the *line* variable limited to 10000 bytes (strictly speaking we can overwrite 10000 bytes reduced by 6 characters of the header ("Host:") and reduced by 96 bytes allocated for *host_name* variable, i.e. 9898 bytes).

Listing 1.1. Vulnerable code in release/src/router/httpd/httpd.c

```
...
void sethost(char *host)
{
    char *cp;

    if(!host) return;

    strcpy(host_name, host);                    //Vulnerable

    cp = host_name;
    for ( cp = cp + 9; *cp && *cp != '\r' && *cp != '\n'; cp++ );
    *cp = '\0';
}
...
static void
handle_request(void)
{
  char line[10000], *cur;
  ...
  /* Parse the rest of the request headers. */
  while ( fgets( cur, line + sizeof(line) - cur, conn_fp ) != (char
      *) 0 )
  {
    ...
    else if ( strncasecmp( cur, "Host:", 5 ) == 0 )
    {
      cp = &cur[5];
      cp += strspn( cp, " \t" );
      sethost(cp);                              //Call to vulnerable
          function
      cur = cp + strlen(cp) + 1;
    }
    ...
  }
  ...
  memset(url, 0, 128);
  if ((query = index(file, '?')) != NULL) {
    file_len = strlen(file)-strlen(query);

    strncpy(url, file, file_len);
  }
  else
  {
    strcpy(url, file);                          //Vulnerable
  }
...
```

Looking at the .bss section further we can see what can be overwritten. We see the *access_ip* variable which stores an eight-byte IP address (IPv4 addresses are of course 4 bytes long but the source code defines the *access_ip* as an array of four integers instead of bytes). Overwriting this value allows one to change the currently logged-in IP address but this is not something that could be further exploited. Next, we have the *delMap* variable, it is used by the websScan function which is actually commented out in *release/src/router/httpd/web.c* so this is also not exploitable. Finally, the *SystemCmd* variable is the last in the .bss section -

the last chance to exploit the heap overflow. The name seems promising so let us verify what is its purpose.

The file *release/src/router/httpd/web.c* defines a kind of Common Gateway Interface which allows one to create dynamically processed responses. It is similar to known solutions, like PHP, JSP, or ASP. It defines several methods which can be used in files with *.asp* extension in order to execute dynamic server actions. The functions executed on the server side are mapped to method names in the *ej_handlers* array (Listing 1.3). One of the handlers is "nvram_dump" method which executes the *ej_dump* function.

Listing 1.2. Dissasembly of .bss section of the httpd binary

```
$ mipsel-linux-objdump -D usr/sbin/httpd
...
Disassembly of section .bss:
...
0046b470 <url@@Base>:
         ...

0046b4f0 <host_name@@Base>:
         ...

0046b550 <access_ip@@Base>:
         ...

0046b588 <delMap@@Base>:
         ...

0046dac4 <SystemCmd@@Base>:
         ...

Disassembly of section .pdr:
...
```

Listing 1.3. Custom CGI handlers defined in release/src/router/httpd/web.c

```
...
struct ej_handler ej_handlers[] = {
...
    { "nvram_dump", ej_dump},
...
}
```

The *ej_dump* function has up to two arguments and if the second argument is not empty it calls the *sys_script* function, as shown on Listing 1.4.

Listing 1.4. ej_dump function declared in release/src/router/httpd/web.c

```
...
static int
ej_dump(int eid, webs_t wp, int argc, char_t **argv)
{
//  FILE *fp;
//  char buf[MAX_LINE_SIZE];
    char filename[PATH_MAX], path[PATH_MAX];
    char *file,*script;
    int ret;

    if (ejArgs(argc, argv, "%s %s", &file, &script) < 2) {
        websError(wp, 400, "Insufficient args\n");
        return -1;
    }

    //csprintf("Script : %s, File: %s\n", script, file);

    // run scrip first to update some status
    if (strcmp(script,"")!=0) sys_script(script);

...
```

Finally, the *sys_script* function is used to execute different statically defined scripts or a custom script provided by the *SystemCmd* global variable - the one we can control. If the second parameter to *sys_script* is "syscmd.sh", it will execute the current *SystemCmd* in a system shell.

Most usages of *sys_script* function are used just after overwriting the *SystemCmd* variable thus overwriting what we could potentially set making the vulnerability not exploitable. However, there are two *CGI* pages that call the "nvram_dump("syscmd.log", "syscmd.sh");" script to present the output of previously executed command. This is exactly what is needed. The pages are:

- Main_Netstat_Content.asp,
- Main_Analysis_Content.asp.

The overwritten value of *SystemCmd* variable will get executed automatically when the authorized user opens one of the two aforementioned above URLs. User interaction is required but the attacker is able to setup a passive "trap" which will await until the administrator logs in and visits the triggering pages. Thanks to the passive nature of such exploit the attack has a great potential.

One may ask what can we do with such an attack, how can the attacker keep router access without forcing the administrator to visit the vulnerable pages over and over again? This can be done thanks to the second vulnerability which the author has discovered: CVE-2017-15656 [3], i.e. ASUS routers store passwords and the whole configuration in plaintext. This allows the attacker to simply read the "http_username" and "http_passwd" properties from nvram in order to access the router with credentials persistently (until the password gets changed).

In order to fully passively exploit the two vulnerabilities we can take one more step. Instead of creating a reverse connection to the attacker's computer (this requires the attacker computer to have a static IP and be online all the time) we can dump the contents of nvram router configuration and store it in

a file which is accessible without authentication. Such files are for example the style-sheets (*.css* files).

The full exploit which sets up a "trap" is a simple curl command as presented in Listing 1.5.

Listing 1.5. The exploit

```
$ curl 'http://routerIP:8080' -H "Host: $(for i in $(seq 1 9700); do
    echo -n " "; done)  \$(nvram show > /www/user/nvram.css )"
```

After triggering the exploit by the administrator the whole nvram dump (including plain text passwords) will be accessible without authentication at:

http://routerIP:8080/user/nvram.css.

4 Post Exploitation Possibilities

Let us assume the attacker gained access to our router. One may ask what are the potential consequences? Unfortunately, depending on the hacker's skills they might be devastating. Because modern routers usually utilize an almost fully functional operating system (usually Linux) - the attacker is able to use its features to perform some further attacks and gain sensitive information. The following subsections will discuss some important examples with brief explanation how the attack can be performed and what is its impact.

4.1 Man-in-the-Middle Attack - DNS

Having administrative access to the router we can force all LAN computers to use different network settings through a change in DHCP configuration. For example it is possible to set up a fake DNS server and redirect users to fake IP addresses in order to perform further MITM attacks. This allows one to dump part of the client's outgoing traffic to some chosen DNS domains. Of course the client will get SSL warnings in the case of the encrypted communication.

4.2 Layer 2 VPN

Some routers have OpenVPN server built-in. For example, the described herein ASUS routers have an OpenVPN installed by default. Having an administrator password allows one to run OpenVPN server.

OpenVPN can be configured to work in TAP mode where it allows one to tunnel OSI layer 2 traffic. This allows the attacker to connect directly to the target LAN. Further, this can be used to attack internal hosts as well as perform MITM and packet sniffing.

4.3 Man-in-the-Middle Attack - SSLStrip

The attacker could also overcome the SSL limits mentioned in Sect. 4.1 by using *SSLStrip* tool [10]. Because this software is written in Python, it would be necessary to convert it into appropriate ELF binary in order to execute it directly on the router. This is possible using the *PyInstaller* or *freeze.py* tool which compiles an ELF Linux binary from python scripts.

Other possible workaround is reconfiguring the DHCP configuration and setting a gateway to the attacker's IP address (utilizing the VPN described in previous section). This way the sslstrip can be executed on attacker's machine.

Having *sslstrip* working in the target network would allow the attacker to sniff critical data, like passwords, even if the traffic was supposed to be encrypted.

4.4 Packet Sniffing

Finally, the attacker could use a packet sniffer running on the router itself in order to dump interesting traffic to file. This could be achieved by compiling *tcpdump* tool with an appropriate toolchain.

In the ASUS routers the system uses a different architecture so a cross-architecture compiler is necessary. Additionally the system does not use standard *libc* but uses its small footprint replacement called *uclibc*. In order to properly link the binary, the *uclibc* toolchain called *buildroot* could be used. Statically linking would be preferred. Finally, coping the *tcpdump* binary to the device and running it allowed one to dump all the traffic flowing through the router. What is worth mentioning - the router is also a LAN L2 bridge, what means that it is possible to sniff also the inter-host traffic inside the LAN.

5 Conclusion

The article discusses the security aspects of modern routers and networking equipment. The observations, however, can be also applied to other embedded devices. The summary of threats was based on an exemplary vulnerabilities disclosed by the author. The article also presented in great detail the vulnerabilities and their exploitation method to show the considerations and risks are real and valid. Finally, the author presents what might be the consequences of a successful attack.

It was shown that sometimes such a simple device as a network router can be a serious threat. Taking control over such router can not only affect this single device but also all the other machines using the internal LAN.

Acknowledgements. The research was supported by Silesian University of Technology grant No. BKM-509/RAU2/2017.

References

1. Adamczyk, B.: CVE-2017-15654. http://sploit.tech/2018/01/16/ASUS-part-II.html. Accessed 17 Jan 2018
2. Adamczyk, B.: CVE-2017-15655. http://sploit.tech/2018/01/16/ASUS-part-I.html. Accessed 17 Jan 2018
3. Adamczyk, B.: CVE-2017-15656. http://sploit.tech/2018/01/16/ASUS-part-II.html. Accessed 17 Jan 2018
4. Artenstein, N.: Broadpwn: Remotely Compromising Android and iOS via a Bug in Broadcom's Wi-Fi Chipsets. In: Blackhat (2017)
5. ASUSTeK Computer Inc.: ASUSWRT. https://www.asus.com/ASUSWRT/. Accessed 17 Jan 2018
6. Bierbaumer, B.: CVE-2017-6548. https://bierbaumer.net/security/asuswrt/#remote-code-execution. Accessed 17 Jan 2018
7. Cutlip, Z.: DLink DIR-815 UPnP Command Injection. http://shadow-file.blogspot.com/2013/02/dlink-dir-815-upnp-command-injection.html. Accessed 17 Jan 2018
8. Fox-IT: CVE-2015-7755. https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2015-7755. Accessed 17 Jan 2018
9. Google Inc.: CVE-2016-10229. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-10229. Accessed 17 Jan 2018
10. Marlinspike, M.: SSLStrip. https://moxie.org/software/sslstrip/. Accessed 17 Jan 2018
11. Postelstorfer, F.: CVE-2014-9583. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-9583. Accessed 17 Jan 2018
12. Reverse Shell Security: routersploit: The Router Exploitation Framework. https://github.com/reverse-shell/routersploit. Accessed 17 Jan 2018
13. Shodan: Shodan Search Engine. https://www.shodan.io/. Accessed 17 Jan 2018
14. Shodan: Vulnerable ASUS Routers - Shodan Report. https://www.shodan.io/report/u7ejeYKQ. Accessed 26 Feb 2018
15. Vanderbeken, E.: TCP-32764. https://github.com/elvanderb/TCP-32764. Accessed 17 Jan 2018

Self-adaptive System for the Corporate Area Network Resilience in the Presence of Botnet Cyberattacks

Sergii Lysenko(✉), Oleg Savenko, Kira Bobrovnikova, and Andrii Kryshchuk

Department of Computer Engineering and System Programming, Khmelnitsky National University, Instytutska, 11, Khmelnitsky, Ukraine
{sirogyk,savenko_oleg_st}@ukr.net, bobrovnikova.kira@gmail.com, rtandrey@rambler.ru
http://ki.khnu.km.ua

Abstract. The paper presents a self-adaptive system for the corporate area networks' resilience in the presence of botnets' cyberattacks. The resilience is ensured by the adaptive reconfiguration of the network. The reconfiguration of the network is carried out based on security scenarios, adopted on the base of the cluster analysis of gathered Internet traffic features inherent to cyberattacks. In order to choose the needed security scenarios, the proposed method uses a semi-supervised fuzzy c-means clustering. Objects of clustering are feature vectors which elements may indicate the appearance of cyber threats in the corporate area networks. The purpose of the technique is to choose security scenarios according to cyberattacks performed by botnets in order to mitigate the attacks and ensure the network's resilient functioning.

Keywords: Botnet · Cyber threat · Cyberattack · Botnet detection
Network defence · Self-adaptive systems · Resilience
Security scenario · Malware · DDoS attack

1 Introduction

The decisive trend in the field of cybersecurity is the transformation of large-scale attacks into a common problem for companies.

Recently multi-vector attacks dominated distributed denial of service (DDoS), reaching nearly 55% of attack types. The most popular multi-vectors combined two vectors, most notably UDP-Flood blended with NTP Amplification, TCP SYN Flood, and ICMP Flood [1].

Nowadays, the main source of the large-scale cyberattacks, including DDoS-attacks, mass spamming and the malware spreading are botnets. A botnet is the network of private computers infected with malicious software and controlled as a group without the owners' knowledge, e.g. to send spam [2].

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 385–401, 2018.
https://doi.org/10.1007/978-3-319-92459-5_31

In the situation of the cyberattacks' implementation, an important task is to assume measures that will able to mitigate the attacks and ensure the stable network's functioning, i.e., the network's resilience.

Resilience is the ability of a server, network, storage system, or an entire data centre, to recover quickly and continue the operation even when there has been an equipment failure, power outage, or other disruption. Specifically, a resilient system is one that can continue to offer a satisfactory level of service even in the face (or in the aftermath of the challenges it experiences [3,4]).

The appearance of new cyber threats, an increase of quantity on the number of malware, as well as the need of the resilient functioning of the networks, require new innovative approaches to provide the efficient information security of computer systems in the corporate area networks [5].

One of the most promising directions that solve the problems of providing the resilience are self-adaptive systems [6]. During its functioning adaptive systems are able to accumulate information for the purpose of assessing changes in external and/or internal conditions and to adapt to these changes by adaptation of their own behaviour. The aim of adaptation can be the improvement of the functionality, efficiency of the system and the insurance of its resilience in conditions of full or partial uncertainty of factors that may have an impact on the adaptive system [7].

This means that the self-adaptive systems can be the base of a new technique for botnet detection and be capable of reacting to known and unknown attacks performed by botnets. Furthermore, it must be able to execute the selfreconfiguration of the network infrastructure depending on type of attack, type of the victim in the network etc.

2 Related Work

2.1 Network Attacks, Methods for Their Detection and Mitigation

Today solutions to the problems of botnet detection and network security are widely presented in the literature.

One of the way to explore the botnets' behaviour is the usage of honeynets. For example, the approaches [8,9] present a detailed analyses of various attacks against the Linuxbased honeypots. The approaches show how to analyze the attacks depending on their types, such as session duration, country and regional Internet registry (RIR) of the attack origin. Furthermore, honeynets are presented as the most efficient tool to detect and analyze threats from the Internet. The authors show recent results for a honeynet made of Dionaea (emulating Windows services), Kippo (emulating Linux services), and Glastopf (emulating website services) honeypots. The disadvantage of the approach is that the differentiation among honeypots according to their IP addresses is relatively rough.

The paper [10] describes an approach of a moving target defence mechanism that defends authenticated clients against Internet service DDoS attacks. The proposed technique involves a group of dynamic, hidden proxies to relay traffic between authenticated clients and servers. In order to defend clients, they

are segregated from malicious cyber intrusions through a series of shuffles, by the means of the continuously replacing attacked proxies with backup proxies and reassigning the attacked clients onto the new proxies. For the purpose of the malicious intrusions segregation implementation, the authors developed an efficient greedy algorithm. In addition, the intrusions' quarantine capability of this greedy algorithm is studied and quantified to enable defenders to estimate resources required to defend against DDoS attacks and meet defined QoS levels under various attacks.

The study [11] proposes a procedure for the information extraction and the botnet detection via footprints of infected system with botnet in order to reconstruct the botnet attack and prepare a digital evidence package which shows the malicious activities and malicious files of this attack to be presented in a court.

In [12] the taxonomies of botnet behavioural features, detection and defence methods are presented. This elevated view highlights opportunities for network defence by revealing shortcomings in existing approaches. In addition, the usefulness of the classification by dimensions for evaluating botnet detection techniques via different metrics of interest is demonstrated. The approach shows the botnet behavioural features and the influence of the taxonomy usage on the accuracy of the botnet detection.

An approach presented in [13] describes principles of botnets functioning from different perspectives: bot candidate selection, network construction, C&C communication mechanisms/protocols, and mitigation approaches. The study provides the mathematical analysis of two P2P botnet elimination approaches: an index poisoning defence and a Sybil defence, and passive monitoring technique based on infiltrated honeypots or captured bots.

The paper [14] presents an approach for both low-rate and high-rate DDoS attacks detection. For this purpose the authors use and empirically evaluate such information metrics as Hartley entropy, Shannon entropy, Renyi's entropy, generalized entropy, KullbackLeibler divergence, and generalized information distance measure. The research involves an appropriate metric that facilitates the construction of an effective model for the low-rate and high-rate DDoS attacks detection.

In [15] a taxonomy of attack tools in a consistent way for the purpose of the network security assuring is provided. The authors also present a comprehensive and structured analysis of existing tools and systems that are able to support both attackers and network defenders. In this approach both pros and cons of the presented systems are discussed. The presented solutions are frequently used in the network infrastructure for assuring security but they are not flexible and are not acceptable for the attacks of zero-days.

In [16] the enterprise network which uses cloud technologies and Software-Defined Networking is analyzed. The authors of the paper studied the impact of security measures on DDoS attack defence mechanisms in an enterprise network. The main idea of the paper is to show that the usage of the DDoS attack mitigation architecture is able to help enterprises to defend against DDoS attacks.

The described architecture integrates a highly programmable network monitoring for enabling the attack detection and a flexible control structure to allow fast and specific reaction to an attack.

In [17] security principles for a wireless network are presented. A clear examination of key security aspects in self-organizing networks and other networks that use wireless technology for the communication is provided. Fundamental security topics and often-used terms are reviewed. After examining critical security issues in a range of wireless networks, specific solutions to security threats are proposed. The main disadvantage of the proposed approach is that it is not able to respond to unknown cyberattacks performed by botnets.

The paper [18] presents an approach for detection of anomalous patterns of network connections using artificial neural networks, immune systems, neuro-fuzzy classifiers, and their combination. The paper describes the architecture of the intrusion detection system based on the proposed technique. The developed intrusion detection system is multi-level: first, a signature-based analysis is carried out, then a combination of adaptive detectors is involved. Performed experiments demonstrate the effectiveness of the methods in terms of false positive, true positive, and correct classification rates.

In [19,20] a technique for detecting network attacks and malicious code is described. The method is based on main principles of artificial immune systems where immune detectors have an artificial neural networks structure. The main goal of the proposed approach is to detect previously unknown cyberattacks. The proposed intelligent cyber defense system can improve the reliability of intrusion detection in computer systems and, as a result, it may reduce financial losses of companies from cyberattacks.

The common weakness of the above listed approaches is that they are not able to adaptively respond to known and unknown attacks performed by botnets nor perform reconfiguration of the network infrastructure depending on the type of a cyberattack to ensure the resilient functioning of the network.

2.2 Botnet Attack Detection in Corporate Area Networks

During the last years, several successful attempts to solve the problem of botnet detection within the corporate area networks (CAN) have been made. Our earlier approaches [21,22] proposed detecting a botnet using a multi-agent system. The botnet detection was performed via agents' communication within a corporate area network. The conclusion about possible botnet's presence was made with the usage of a fuzzy system that takes into account the information about the botnet demonstrations obtained from the several computer systems of the networks.

The further development of our botnet detection approaches involved DNS-based techniques [23]. The botnet detection was executed via the passive DNS-monitoring and active DNS probing in the network. This made it possible to identify botnets which used such evasion techniques as cycling of IP mapping, "domain flux", "fast flux", DNS-tunneling. The described technique has become the base of our BotGRABBER LAN tool, able to gather the DNS-traffic and analyze features obtained from the payload of DNS-messages, which indicated

the usage of evasion techniques by botnets. The conclusion about the possible botnet's presence was made with the usage of the clustering analysis.

Our approach presented in [24] enunciated the evolution of a botnet detection system – BotGRABBER. It was enhanced by the possibility of localization of botnets in a CAN by the means of not only the DNS-traffic analysis but also the analysis of behaviour of the malicious software in the hosts of the network. Now, the BotGRABBER system is able to monitor and analyze the DNS-traffic which allows one to make conclusion about network hosts' infections with a bot of the botnet. Nevertheless, even having the opportunity to detect threats and localize them, the described system is unable to assure the resilient functioning of the network in the situation of a botnet attack. The problem is that such threats not only slow down the network but may also cause the unavailability of network services and the whole infrastructure that cannot cope due to capacity insufficiency. Given this, the following open questions arise: What to do in the situation when a botnet is producing or has already executed cyberattacks and the network is unstable? How to respond to different situations in the presence of botnet cyberattacks coming from the inside or outside of the network? And the major question is: What to do if the cyberattack is unknown to the antiviral tools?

An answer can be a proposal to construct a system that will be able to respond adaptively to threats caused by bots of a botnet. Such respond consists in choosing the needed security scenario of network's reconfigurations to avoid or mitigate the influence of botnet's attacks and to provide the resilient functioning of the network.

3 Self-adaptive System for the Corporate Area Networks Resilience in the Presence of Botnet Cyberattacks

We propose a self-adaptive system for assuring the corporate area networks' resilience in the presence of botnets' cyberattacks. The resilience is ensured by the adaptive reconfiguration of the network. This reconfiguration is carried out based on a security scenario, adopted on the base of the cluster analysis of previously gathered features inherent to the cyberattacks. The features are formed as feature vectors and are to be clustered. A result of the clustering is the assignment of each feature vector to a cluster which is corresponding to a given cyberattack. The purpose of the technique is the choice of the needed security scenario of the network reconfiguration according to the cyberattacks performed by botnets.

The proposed approach includes several stages (Fig. 1).

1. The learning stage consists of the following steps:

1.1. Knowledge formation based on the features that may indicate cyberattacks performed by a botnet;

1.2. Presentation of the knowledge about the cyberattacks as a set of feature vectors;

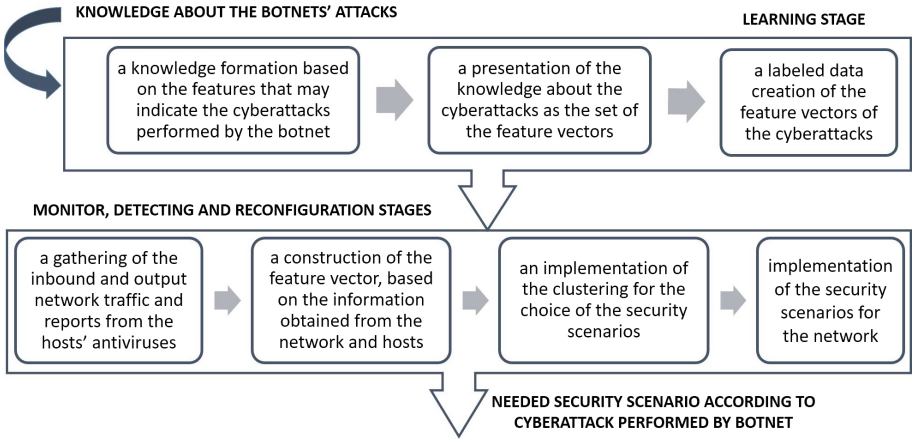


Fig. 1. Functioning process of the self-adaptive system for the corporate area networks resilience in the presence of botnet cyberattacks

1.3. Labeling the obtained feature vectors of the cyberattacks for the purpose of clusters' formation, where each cluster corresponds to some cyberattack, and in turn to some security scenario, which is to be applied for the cyberattack mitigation.

2. The monitoring stage consists of the following steps:

2.1. Gathering the inbound and outbound network traffic and gathering the information about the hosts' network activity and reports of the hosts' antiviruses;

2.2. Construction of the feature vectors based on the information obtained from the network and hosts.

3. The detecting stage involves the implementation of the semi-supervised fuzzy c-means clustering of the obtained feature vectors in order to assign them to one of the clusters and choosing the proper security scenario.

4. The reconfiguration stage involves the implementation of the security scenario for the corporate area network's infrastructure.

Let us discuss each step of the proposed technique in detail.

The knowledge formation about the features that may indicate a cyberattack performed by a botnet. Let us denote the set of network components, that can be attacked by botnet, as $B = \{b_1, b_2, b_3\}$, where b_1 – network host, b_2 – network device, b_3 – server in the network, $b_i \in B$.

Then let us denote the set of cyberattacks, performed by botnet, as $A = \{a_j\}_{j=1}^{N_A}$, where a_1 – the botnet attack performed by a worm-virus; a_2 – the botnet attack performed by malware (for instance, Trojan programs, Backdoors, etc.); a_3 – the ping flooding attack; a_4 – the smurf attack; a_5 – the demonstration of the TCP SYN Flood attack; a_6 – the Fragmented UDP Flood attack; a_7 – the DNS Amplification attack; a_8 – the TCP Reset attack; a_9 – the ICMP Flood attack; a_{10} – the SIP INVITE Flood attack; a_{11} – the Encrypted SSL DDoS

attack; a_{12} – the ping sweep attack; a_{13} – the SQL/PHP injection attack; a_{14} – the Cross-Site Scripting attack; a_{15} – the phishing attack; a_{16} – the DNS spoofing attack; a_{17} – the ping of death attack; a_{18} – the R-U-Dead-Yet DDos attack (R.U.D.Y.) [25], N_A – the number of cyberattacks. The features, which are to be analyzed to identify the above-mentioned cyberattacks [23,26,27] are presented in Table 1. Let us denote the set of security scenarios as $S = \{s_m\}_{m=1}^{N_S}$, where N_S – the number of security scenarios that are to be applied depending on the type of an attack performed by a botnet. Thus, the function of the choice of a security scenario for network reconfiguration in the presence of a specified type of a botnet attack f can be presented as: $f: b_i \times a_j \rightarrow s_m$. Presentation of the knowledge about the cyberattacks as the set of feature vectors. All the above-mentioned features are the base of the set of feature vectors $X = \{x_k\}_{k=1}^{N}$, where each of feature vector x_k describes the cyberattack or its absence, N – the number of the feature vectors. Using the features presented in the feature vectors, the set of rule R is built in order to describe each cyberattack. The set of feature vectors forms the training set, which is used for the semi-supervised learning.

An example of rule R_{b_2}, which describes the attack against the network hosts and makes the choice of the needed security scenario, is the following:

$$\begin{aligned} &if\ ((l_N \in [75,255]\ and\ n_U \in (27,37])\ or\ (e_N \geq f_{Eb32}\ or\ (e_R \geq f_{Eb64}\ or \\ &\quad or\ e_R \geq f_{Eb256})\ or f_{UR} = 1))\ and\ l_P > 300 \Rightarrow a_2 \Rightarrow s_2 \end{aligned} \tag{1}$$

NOTE. If different values concerning the same attack are observed, the system produces different security scenarios. For example, the same attack may be acted against the server or against the network hosts. In this case, different rules R_{b_i} will be used and different security scenarios will be applied.

Labeling the obtained feature vectors of the cyberattacks for the purpose of clusters' formation. Let c denote the number of the predefined clusters of feature vectors. Each cluster corresponds to the specified cyberattacks (and the security scenario to be applied) and one cluster corresponds to the absence of the attack.

The membership of the feature vector x_k to the i-th cluster indicates the presence or absence of a cyberattack and the need to apply or not to apply the security scenario, respectively.

In order to build the centroid (the prototype) of the i-th clusters, v_i, we have to assume the labeled data. The labeled data is based on knowledge of the features which may indicate the botnet attacks in the network and is presented as the set of feature vectors. Each feature vector x_k of labeled data belongs to one of the predefined clusters.

The semi-supervised fuzzy c-means clustering is based on the minimization of the following objective function [28]:

$$J_k = \sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^{p} d_{ik}^{2} + \alpha \sum_{i=1}^{c}\sum_{k=1}^{N} (u_{ik} - f_{ik} b_k)^{p} d_{ik}^{2}, \tag{2}$$

where N – the total number of the feature vectors to be clustered (labeled and unlabeled feature vectors), u_{ik} – the membership value for the k-th feature

Table 1. The features analyzed to identify the cyberattacks

Feature	Description
l_P	An average payload length per connection
n_{PZ}	A number of a different size of packets transferred to a total number of frames per connection
b_{EH}	A total number of bytes per connection excluding the header
b_{TC}	A total number of bytes transmitted per connection
d_C	A duration of the connection
b_{OD}	A number of bytes transmitted from origin to destination
p_{OD}	A number of packages transmitted from origin to destination
p	Transmission protocol
f_{IO}	A boolean feature that indicates whether the inbound traffic has an associated outbound traffic record
d_{EL}	A duration of the connection, observed from the earliest of the associated inbound or outbound traffic until the end of the latter traffic
b_S	Total size for the session in bytes
p_S	Total number of packets in the session
o_{SS}, i_{SS}	Self-similarity of the outbound/inbound packets in the session, determined by examining the variance in size of the outbound/inbound packets using the Hurst exponent
v_{OPS}, v_{IPS}	Velocity of outbound/inbound traffic measured in packets per second
v_{OBS}, v_{IBS}	Velocity of outbound/inbound traffic measured in bits per second
v_{OBP}, v_{IBP}	Velocity of outbound/inbound traffic measured in bytes per packet
d_{PS}	Standard deviation of packet size within the session measured in bytes
f_{TCP}	Invalid values of TCP flags seen in this session
f_{GEO}	The geolocation feature defined by IP-address
s_{RT}	Server response time, milliseconds
n_{ARP}	A number of the ARP-requests
r_{NAT}	A number of records in the NAT/PAT-table
m_R	A size of the router's memory used, megabytes
p_R	A value of the router's processor's time, %
n_{DP}	An amount of denied packets
l_N	The length of the domain name
n_U	The number of unique characters in the domain name
e_N	Entropy of the domain name
t_{mod}, t_{med}, t_{aver}	TTL-periods (mode, median, average value)
n_A	The number of A-records corresponding to the domain name in the incoming DNS-message
n_{IP}	The number of IP-addresses concerning the domain name
s_{IP}	The average distance between the IP-addresses concerning domain name
s_A	The average distance between the IP-addresses in the set of A-records for domain name in the incoming DNS-message
n_{UA}	Number of unique IP-addresses in sets of A-records corresponding to the domain name in the DNS-messages
s_{UA}	The average distance between unique IP-addresses in sets A-record corresponding to the domain name in the DNS-messages
n_D	Number of domain names that share IP-address corresponding to the domain name
f_{UR}	The sign of the usage of uncommon types of the DNS records, or DNS records that are not commonly used by a typical client, 0 – if there is such usage, 1 – otherwise
e_R, f_{Eb}	Entropy of the DNS-records, which are contained in the DNS-messages
l_P	Maximum size of the DNS-messages about domain name
f_S	The sign of success of DNS-query

vector in the i-th cluster, f_{ik} – the membership value of the k-th labelled feature vector in the i-th cluster, d_{ik} – the distance between the k-th feature vector and prototype of the i-th cluster, $b = [b_k]$ – a boolean indicator, which distinguishes the labeled and unlabeled feature vectors:

$$b_k = \begin{cases} 1, & \text{if feature vector } x_k \text{ is labeled,} \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

The centroid of the i-th cluster, v_i, and the partition matrix u_{ik} are presented in the Eq. (4) [28]:

$$v_i = \frac{\sum_{k=1}^{N} u_{ik}^2 x_k}{\sum_{k=1}^{N} u_{ik}^2}, \quad u_{ik} = \frac{1}{1+\alpha} \left\{ \frac{1 + \alpha(1 - b_k \sum_{l=1}^{c} f_{lk})}{\sum_{l=1}^{c} \left(\frac{d_{ik}}{d_{lk}} \right)^2} + \alpha f_{ik} b_k \right\}, \tag{4}$$

where α denotes a scaling factor to maintain a balance between the supervised and unsupervised component within the optimization mechanism [28].

As a distance metric between the k-th feature vector and the centroid of cluster the Mahalanobis distance was used:

$$d_{ik} = \|x_k - v_i\|^T A \|x_k - v_i\|, \tag{5}$$

with A being a positive definite matrix in $R^n \times R^n$.

Gathering the inbound and outbound network traffic and the information about the hosts' network activity and reports of the hosts' antiviruses. At this stage of our method for the purpose of the network-type cyberattacks detection, the monitoring of the network activity, that may indicate the appearance of a cyberattack, is performed. For the purpose of the host-type cyberattack detection, the information about the hosts' network activity and the reports of the hosts' antiviruses is collected. The gathered information is sent to the classifier for the further analysis.

Construction of the feature vectors based on the information obtained from the network and hosts and the implementation of the semi-supervised fuzzy c-means clustering for the security scenarios choice. The data gathered at the previous stage is then analyzed. The result of the analysis is conclusion about the presence or absence of an attack and the corresponding security scenario for network reconfiguration. As the means of the security scenario choice the semi-supervised fuzzy c-means clustering was used. At the detecting stage the objects of the clustering are the feature vectors x_k, obtained in the analysis of the payload of the inbound and outbound traffic, as well as the reports of the antiviral tool about the possible hosts' infection. The result of clustering are the membership values u_{ik} of the feature vector x_k to each cluster i. The membership of feature vector x_k to the i-th cluster provides the scenario of the network reconfiguration, which is to be applied in the situation of a botnet's attack.

Implementation of the security scenarios for the corporate area network's infrastructure. Based on the choice, made at the previous stage the security scenario is to be applied. Each scenario contains the list of actions of the network reconfiguration.

For example, to mitigate the effects of R.U.D.Y. DDoS attack on the server, the possibly chosen security scenario involves the following actions:

- the decrease of the amount of time the server will wait for certain events before failing a request;
- the decrease of the amount of time the server will wait for subsequent requests on a persistent connection;
- the decrease of the clients' request sizes: the size of the HTTP request header allowed from the client, the number of HTTP request header fields that will be accepted from the client, and the size of the HTTP request line that will be accepted from the client;
- the decrease of the maximum size of the upload files: the total size of the HTTP request body sent from the client;
- the size of an XML-based request body;
- the decrease of the maximum amount of requests that can be served simultaneously, with any number going past the limit being queued (the restriction of the number of clients);
- the decrease of the maximum size of the data sent to the server: the maximum POST data size and the decrease of the size of POST data excluding the file uploads;
- the decrease of the amount of the "request body" data (POSTed data), which should be kept in the memory (RAM);
- blocking of the traffic from host-name and IP addresses, which are the source of the malicious traffic.

NOTE. The system also includes several security scenarios which contain recommendations of the already infected systems' reinstallation. In majority of cases, a botnet infection cannot be entirely removed automatically and network's or hosts' reconfiguration is inefficient. This may occur, e.g., in the situation of pandemic attacks, such as Petya.A [29].

4 Experiments

A number of experiments were carried out to determine the effectiveness of the proposed method. In the experiments a local area network of 50 hosts (each one with Microsoft Windows operating system), one dedicated server (Linux OpenSusE operating system with nginx HTTP server) were used. The experiments lasted 24 h. Network traffic was captured by the means of tcpdump utility. All functionality of the proposed technique was implemented into the BotGRABBER system, described in Sect. 2.2.

During the experiments 150 attacks of different types against the hosts, server and routers were performed. The aim was to find out whether the corporate area

network will be able to function in the situation of the attacks (for example, if the server, hosts or network router will be able to respond in specified time). In this paper detailed results of the experiments with the R.U.D.Y., smurf, and MAC flooding attacks [25] are discussed.

The R.U.D.Y. attack on a network server is a low-level and slow attack tool, designed to crash a web server by submitting long form fields. The attack is executed via a DoS tool which browses the target website and detects embedded web forms. Once the forms have been identified, R.U.D.Y. sends a legitimate HTTP POST requests with an abnormally long "content-length" header field and then it starts injecting the form with information, one byte-sized packet at a time. Such a type of attacks is difficult to detect compared to volumetric DDoS attacks which are noticeable due to the abnormally high fluctuation in incoming traffic. In order to perform this attack, the R.U.D.Y. simulation tool [30] was used.

The smurf attack is characterized by a large number of ICMP packets with the victim's spoofed source IP to a network using an IP broadcast address. This causes devices in the network to respond by sending a reply to the source IP address. In order to perform the smurf attack, the Hyenae network packet generator [31] was used.

The MAC flooding attack possess a finite hardware learning table to store the source addresses of all received packets: when this table becomes full, the traffic that is directed to the addresses that cannot be learned anymore will be permanently flooded. In order to perform the MAC flooding attack the Dsniff tool (macof) [32] was used.

Figures 2, 3 and 4 demonstrate values of traffic rates and server response times before an attack, during the attack and after the security scenario's implementation. Thus, we can see, that during the R.U.D.Y. attack the traffic rates remain almost unchanged (Fig. 2a), but the server response times increased, that caused the service unavailability (Fig. 2b). The appliance of the security scenario, produced by the BotGRABBER system, affected the slightly visible changes in the traffic rates, while the server response times decreased and server was able to respond. During the smurf attack both the rates of traffic and the server response times increased greatly. The implementation of the security scenario, produced by the BotGRABBER system, affected significantly the traffic rates (Fig. 3a), while the server response times decreased to the normal level and server was also able to respond (Fig. 3b).

On the other hand, the experiments with the involvement of the MAC flooding attack not only demonstrated the increasing rates of traffic and response times, but also caused the situation when the system demonstrated the impossibility of the automated network reconfiguration (Fig. 4). In this situation, we had to perform the mitigation measures manually.

In order to assess the assurance of the network's resilience, the integrated resilience metric presented in [33] was used:

$$GR = R \times \left(\frac{RAPI_{RP}}{RAPI_{DP}} \right) \times (TAPL)^{-1} \times RA, \tag{6}$$

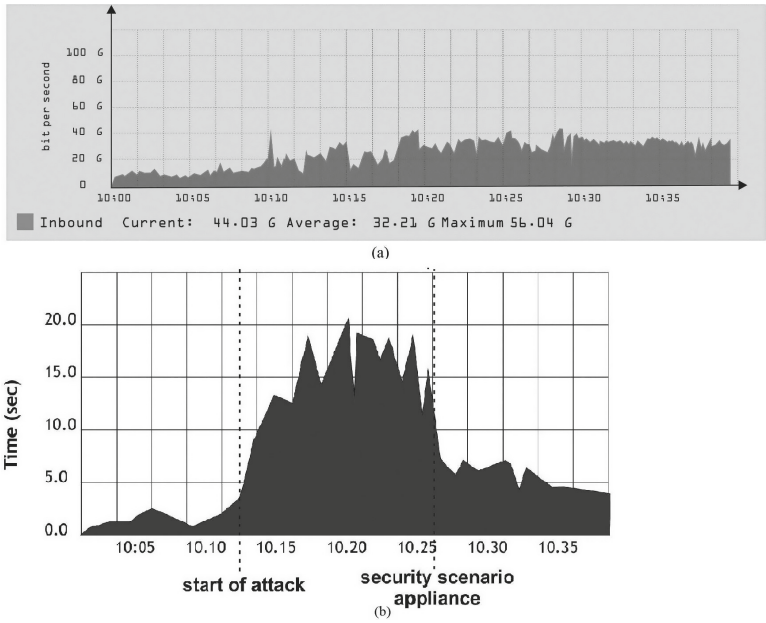


Fig. 2. Traffic rates (a) and server response times (b) before, during, and after the R.U.D.Y.-attack

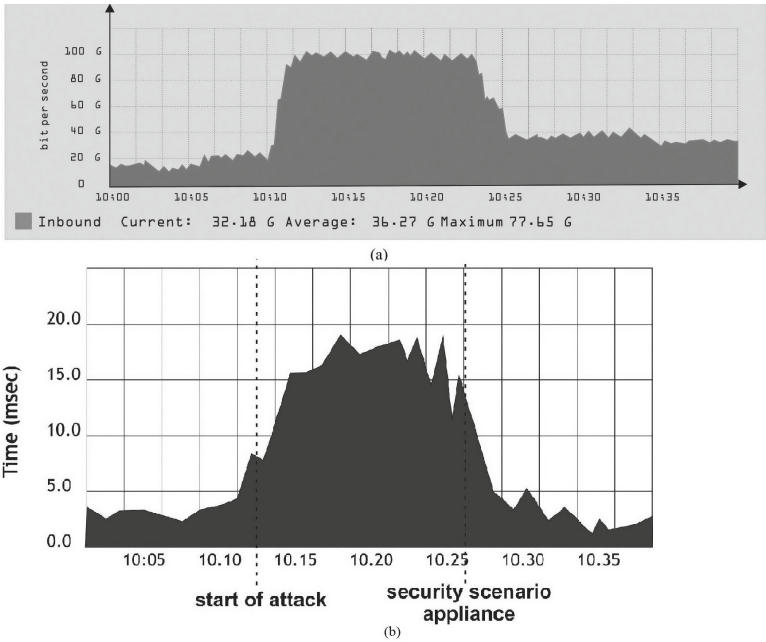


Fig. 3. Traffic rates (a) and server response times (b) before, during, and after the smurf attack

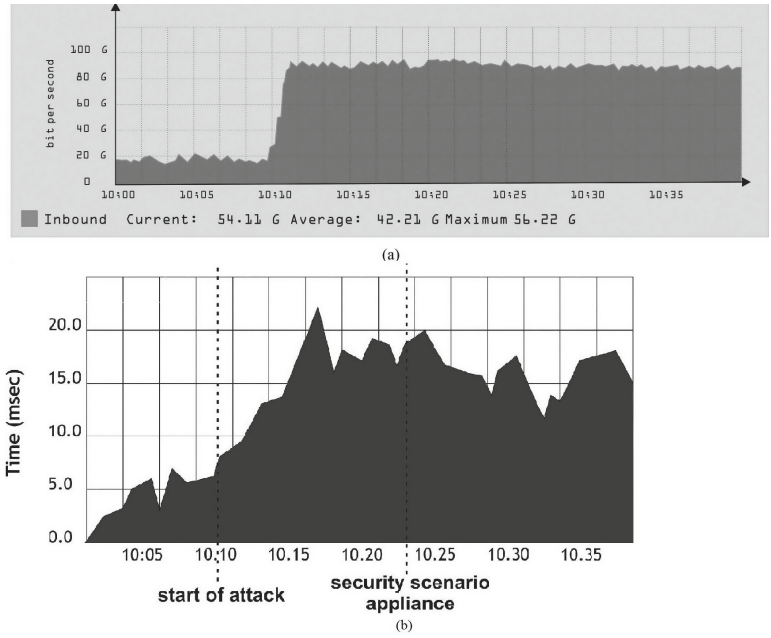


Fig. 4. Traffic rates (a) and server response times (b) before, during, and after the macflooding attack

where R – robustness (or resistance), which is a measure to assess the capability and quantifies the minimum MOP value between t_d and t_{ns}, MOP is the measure of the system (network) performance, which is normalized between 0 and 1 (where 0 is a total loss of operation and 1 is the target MOP value during the normal network functioning); t_d represents the time when the network is under a botnet attack and t_{ns} represents the time when the network was reconfigured based on the proposed security scenario; $RAPI_{DP}$ – rapidity during the phase of the botnet attack, which can be approximated by the average slope of the MOP function; $RAPI_{RP}$ – rapidity during the network reconfiguration phase; $TAPL$ – time-averaged performance loss, which considers the time of appearance of a botnet attack up to network reconfiguration; RA – reconfiguration ability, a quantitative measure, that describes the system performance, reached after the appliance of the security scenario.

The selection of the appropriate MOP depends on the specific service provided by the infrastructure under analysis.

We consider reconfigurations of the network during an attack successful if $GR > \delta$, where δ is the predefined threshold. Taking the resilience metric into account, the rates of the successful network reconfigurations that leads to the mitigation of the attacks are presented in Table 2. Thus, the involvement of the

adaptive system into the BotGRABBER demonstrates the ability to ensure the resilient network functioning in the situation of the cyberattacks by botnets at the rate of 70%.

Table 2. Summary of the results of the experiments

The target of an attack'	Number of attacks	Number of successful reconfigurations
Network hosts	50	41
Server	50	38
Routers	50	36
Total	150	105

5 Discussion

In spite of promising results of the experiments, there are some recommendations for the better BotGRABBER's functioning. In order to assure its efficient operation the first task is to build correct security scenarios. The problem is that security scenarios construction depends on such elements as a network architecture, network devices, hosts' operating systems, server's properties and features, etc. All these factors influence greatly on the content of the security scenarios. That is why in further research it is required to include into BotGRABBER a possibility to choose groups of security scenarios depending on the aforementioned network components with its software.

At present time, the main weakness of the approach is that the BotGRABBER system is not able to detect all of cyberattacks and therefore, it does not comprehends the appropriate security scenarios. That is because some of attacks are multi-vector and in some cases it is impossible to build all attack's variants. However, the usage of the fuzzy clustering based on the knowledge about botnets' attacks enables one to conclude about possible unknown attacks and therefore, to propose the security scenario for mitigation of the influence of botnet's attacks and assurance of the resilient functioning of the network. Another aspect of BotGRABBER system functioning is that there is a possibility of a situation when security scenarios may contain recommendations of the manual adjustment instead of the automated network reconfiguration (the necessity to contact the system administrator).

Summing up, the developed technique demonstrates acceptable results for the assuring the network's resilient functioning in the presence of the botnets' cyberattacks. Future work should involve such improvements as new knowledge acquisition about known and new cyberattacks, which will be able to eliminate the present problems.

6 Conclusions

The paper presents the self-adaptive system for the corporate area networks' resilience in the presence of botnets' cyberattacks. The resilience is ensured by the adaptive reconfiguration of the network. The answer to the question how the network has to be reconfigured is received by the means of the cluster analysis of the cyberattacks' features, which are observed in the network and the network hosts. In order to choose the needed security scenario, the proposed method uses the semi-supervised fuzzy c-means clustering. Objects of clustering are the vectors of traffic features, which may indicate the appearance of cyber threats in a corporate area network.

The purpose of the technique is to choose the network and network hosts' reconfiguration scenarios according to cyberattacks performed by botnets. Usage of the developed system makes it possible to detect a high percentage of known and unknown multi-vector cyberattacks performed by the botnets. Experimental results demonstrated that the proposed technique ensures the resilient network functioning in the situation of the cyberattacks by botnets at the rate of about 70%.

Acknowledgments. We thank the Khmelnytskyi National University for providing access to local network during the performance of the experimental research.

References

1. NEXUSGUARD: DDoS Threat Report 2017 Q3. https://www.nexusguard.com/threat-report-q3-2017
2. Oxford Dictionaries. http://www.oxforddictionaries.com/definition/english/botnet?q=botnet
3. SearchDataCenter. Data Center Resiliency. http://searchdatacenter.techtarget.com/definition/resiliency
4. Giudice, M., Wilkinson, C.: Crowe Horwath: Resilience Going Beyond Security to a New Level of Readiness (2016). https://www.crowehorwath.com/insights/asset/cyber-resilience-readiness-level
5. Knapp, E.D., Langill, J.T.: Industrial Network Security: Securing Critical Infrastructure Networks for Smart Grid, SCADA, and Other Industrial Control Systems, vol. 460. Syngress (2014)
6. Cheng, B.H.C., et al.: Software engineering for self-adaptive systems: a research roadmap. In: Cheng, B.H.C., de Lemos, R., Giese, H., Inverardi, P., Magee, J. (eds.) Software Engineering for Self-Adaptive Systems. LNCS, vol. 5525, pp. 1–26. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02161-9_1
7. Macías-Escrivá, F.D., Haber, R., Del Toro, R., Hernandez, V.: Self-adaptive systems: a survey of current approaches, research challenges and applications. Exp. Syst. Appl. **40**(18), 7267–7279 (2013)
8. Zuzcak, M., Sochor, T.: Behavioral analysis of bot activity in infected systems using honeypots. In: Gaj, P., Kwiecień, A., Sawicki, M. (eds.) CN 2017. CCIS, vol. 718, pp. 118–133. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59767-6_10

9. Sochor, T., Zuzcak, M.: Attractiveness study of honeypots and honeynets in internet threat detection. In: Gaj, P., Kwiecień, A., Stera, P. (eds.) CN 2015. CCIS, vol. 522, pp. 69–81. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19419-6_7
10. Wang, H., Jia, Q., Fleck, D., Powell, W., Li, F., Stavrou, A.: A moving target DDoS defense mechanism. Comput. Commun. **46**, 10–21 (2014)
11. Javadianasl, Y., Manaf, A.A., Zamani, M.: A practical procedure for collecting more volatile information in live investigation of botnet attack. In: Hassanien, A.E., Fouad, M.M., Manaf, A.A., Zamani, M., Ahmad, R., Kacprzyk, J. (eds.) Multimedia Forensics and Security. ISRL, vol. 115, pp. 381–414. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-44270-9_17
12. Khattak, S., Ramay, N.R., Khan, K.R., Syed, A.A., Khayam, S.A.: A taxonomy of botnet behavior, detection, and defense. IEEE Commun. Surv. Tutorials **16**(2), 898–924 (2014)
13. Wang, P., Wu, L., Aslam, B., Zou, C.C.: Analysis of Peer-to-Peer botnet attacks and defenses. In: Król, D., Fay, D., Gabryś, B. (eds.) Propagation Phenomena in Real World Networks. ISRL, vol. 85, pp. 183–214. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-15916-4_8
14. Bhuyan, M.H., Bhattacharyya, D.K., Kalita, J.K.: An empirical evaluation of information metrics for low-rate and high-rate DDoS attack detection. Pattern Recognit. Lett. **51**, 1–7 (2015)
15. Hoque, N., Bhuyan, M.H., Baishya, R.C., Bhattacharyya, D.K., Kalita, J.K.: Network attacks: taxonomy, tools and systems. J. Netw. Comput. Appl. **40**, 307–324 (2014)
16. Wang, B., Zheng, Y., Lou, W., Hou, Y.T.: DDoS attack protection in the era of cloud computing and software-defined networking. Comput. Netw. **81**, 308–319 (2015)
17. Pathan, A.S.K. (ed.): Security of Self-organizing Networks: MANET, WSN, WMN, VANET, vol. 638. CRC Press, Boca Raton (2016)
18. Branitskiy, A., Kotenko, I.: Network attack detection based on combination of neural, immune and neuro-fuzzy classifiers. In: 2015 IEEE 18th International Conference on Computational Science and Engineering (CSE), pp. 152–159 (2015)
19. Komar, M., Sachenko, A., Bezobrazov, S., Golovko, V.: Intelligent cyber defense system using artificial neural network and immune system techniques. In: Ginige, A., et al. (eds.) Information and Communication Technologies in Education, Research, and Industrial Applications, ICTERI 2016. CCIS, vol. 783, pp. 36–55. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69965-3_3
20. Bezobrazov, S., Sachenko, A., Komar, M., Rubanau, V.: The methods of artificial intelligence for malicious applications detection in Android OS. Int. J. Comput. **15**(3), 184–190 (2016)
21. Lysenko, S., Savenko, O., Kryshchuk, A., Kljots, Y.: Botnet detection technique for corporate area network. In: Proceedings of the 2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS), pp. 363–368 (2013)
22. Pomorova, O., Savenko, O., Lysenko, S., Kryshchuk, A.: Multi-agent based approach for botnet detection in a corporate area network using fuzzy logic. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2013. CCIS, vol. 370, pp. 146–156. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38865-1_16

23. Pomorova, O., Savenko, O., Lysenko, S., Kryshchuk, A., Bobrovnikova, K.: Anti-evasion technique for the botnets detection based on the passive DNS monitoring and active DNS probing. In: Gaj, P., Kwiecień, A., Stera, P. (eds.) CN 2016. CCIS, vol. 608, pp. 83–95. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39207-3_8
24. Lysenko, S., Savenko, O., Bobrovnikova, K., Kryshchuk, A., Savenko, B.: Information technology for botnets detection based on their behaviour in the corporate area network. In: Gaj, P., Kwiecień, A., Sawicki, M. (eds.) CN 2017. CCIS, vol. 718, pp. 166–181. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59767-6_14
25. IMPERVA INCAPSULA. https://www.incapsula.com/ddos/attack-glossary
26. Najafabadi, M.M., Khoshgoftaar, T.M., Napolitano, A., Wheelus, C.: RUDY Attack: detection at the network level and its important features. In: FLAIRS Conference, pp. 288–293 (2016)
27. Alejandre, F.V., Cortés, N.C., Anaya, E.A.: Botnet detection using clustering algorithms. Res. Comput. Sci. **118**, 65–75 (2016)
28. Pedrycz, W., Waletzky, J.: Fuzzy clustering with partial supervision. IEEE Trans. Syst. Man Cybernet. Part B (Cybernet.) **27**(5), 787–795 (1997)
29. VIRUS BULLETIN. Grooten, M.: VB2017 Videos on Attacks Against Ukraine. https://www.virusbulletin.com/blog/2017/12/vb2017-videos-attacks-against-ukraine/
30. SOURCE FORGE: R-U-Dead-Yet? (RUDY) Original Source Code Files. https://sourceforge.net/projects/r-u-dead-yet/
31. SOURCE FORGE: Hyenae. https://sourceforge.net/projects/hyenae/
32. dSniff. https://www.monkey.org/~dugsong/dsniff
33. Linkov, I., Palma-Oliveira, J.M. (eds.): Resilience and Risk: Methods and Application in Environment, Cyber and Social Domains, vol. 580. Springer, Dordrecht (2017). https://doi.org/10.1007/978-94-024-1123-2

QoS- and Energy-Aware Services Management of Resources in a Cloud Computing Environment

Jerzy Martyna(✉)

Institute of Computer Science, Faculty of Mathematics and Computer Science, Jagiellonian University, ul. Prof. S. Lojasiewicza 6, 30-348 Cracow, Poland
martyna@ii.uj.edu.pl

Abstract. Cloud computing systems are playing an increasingly important role in business, consumer and scientific domains. However, data centres in which huge amounts of data are collected, contribute to gigantic carbon dioxide emissions. Moreover, improving the quality of the service (QoS), described with QoS parameters, results in additional emissions of carbon dioxide into the atmosphere. The main purpose of the solution presented is to minimise the consumption of electricity and guarantee all users the required QoS. The proposed solution based on the proposed heuristic is close to the optimal one. The simulation test confirms that the overall performance is satisfactory, which confirms the applicability of this solution in practice.

Keywords: Clouds · Resource allocation · Analytical models
Energy efficiency

1 Introduction

Cloud computing is an emerging computing paradigm which can significantly change the way information is processed around the world. It is one of the three leading IT technologies. Otherwise, cloud computing is an important step in technological progress that will provide the impetus for the development of new software and hardware solutions for all corporate, individual and scientific users. It is recalled that they enable a single user to be provided with on demand computing resources that are sufficient for his needs among all computing resources while being reduced in computing cost by the increasing infrastructure utilization. The principle of cloud computing lies in transferring the entire burden of providing resources, such as data, software or computing power, to the server and enabling constant access through client computers. The main enabling technology for cloud computing is virtualisation. Thanks to software virtualisation there is separation and physical computing device into one or more "virtual" devices, with each of them can be used and managed to perform computing tasks. Then idle computing resources can be allocated for other users, which makes the computing grid more efficient [12,15].

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 402–413, 2018.
https://doi.org/10.1007/978-3-319-92459-5_32

Cloud computing is an important step in the technological progress that will provide impetus for the development of new software solutions [4] for all corporate, individual and scientific users. This will cause a rapid development of both applications and computing centres that will act as service providers. A necessary infrastructure condition for the creation of such data centres is an appropriate network infrastructure, guaranteeing adequate data flows that do not contribute to delays. In addition, the costs of their maintenance and servicing must be acceptable to the owners. It is anticipated that cloud computing can improve customer engagement and operational performance while growing market and macro-economic factors.

The business potential of cloud computing is recognised by several IT companies. For example, one of the largest cloud computing services is Google Cloud Platform developed by Google Inc. [11]. The Aneka cloud (commercialised by Manjrasoft) [22] is used for developing, deploying, and managing cloud applications. Aneka consists of a scalable cloud middleware that can be deployed on top of heterogeneous computing resources. It offers an extensible collection of services coordinating the execution of applications, helping administrators monitor the status of the cloud, and providing integration with existing cloud technologies. The largest cloud computing is the Amazon Elastic Compute Cloud [1]. It provides scalable computing capacity in the Amazon Web Services [2] cloud.

Noteworthy are the implementations of Green Cloud computing, which aim to reduce carbon dioxide (CO_2) emissions. Among other things, the Japanese government recommended a significant reduction in electricity consumption by the Japan Data Centre Council in various data centres [16]. Additionally, the world's leading computing service providers also came up with the initiative to set up a global consortium known as the Green Grid [10] to reduce pollution emitted into the atmosphere. At one time it had more than 175 member companies.

One of the models of cloud computing is the service-oriented model [21]. In this model, it is assumed that cloud computing consists of three layers. The first layer is the infrastructure as a service (IaaS), the second layer is the platform as a service (PaaS), and the third layer is software as a service (SaaS). If the model is oriented towards saving energy, a better model is the high-level architecture for supporting energy-efficiency in Green Cloud computing [5]. This model details several layers (see Fig. 1). The first layer consists of users and their brokers. They can also be a company deploying Web applications. Each of them formulates their demands for sharing resources of cloud computing. The second layer is the Green Resource Allocator, which is responsible for the service requirements of submitted requests before deciding whether to accept or reject them. This layer includes consumer and cloud interfaces. In addition, this layer is responsible for the customer's energy consumption profile, energy consumption monitor, service scheduler etc. This layer sets the price for using the cloud. The third layer is Virtual Machines (VMs). This layer is responsible for the placement of virtual machines in the fourth layer. The fourth (final) layer is the physical computer nodes, which are typically computing servers.

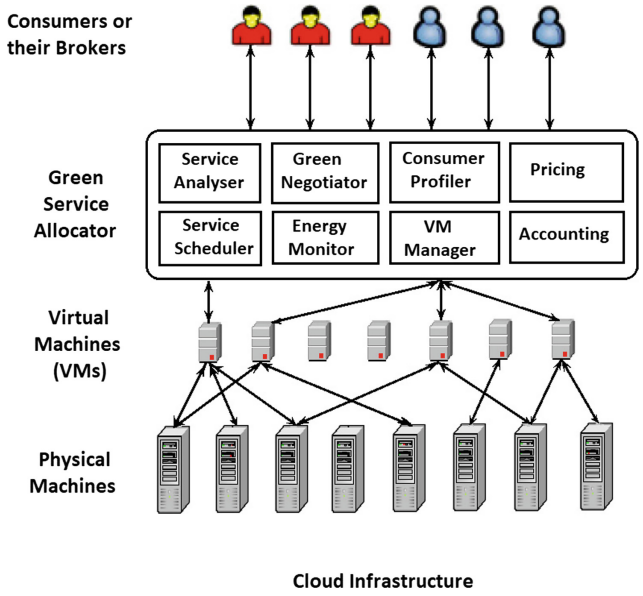


Fig. 1. An architecture of layered cloud computing infrastructure.

An important problem associated with the design of cloud computing systems is the minimization of server power consumption. Even a completely idle server consumes about 70% of its peak demand [9]. Therefore, keeping unused computing power leads to excessive use of electricity. Furthermore, maintaining as operating cooling systems also requires the use of electricity. According to a scientific report [18], for each watt of power consumed by computers, an additional 0.5−1 W is required. It is clear that high energy consumption leads to substantial carbon dioxide (CO_2) emissions, providing a greenhouse effect.

The second important problem next to energy consumption that must be solved in cloud computing design is the performance of the whole system. If the cloud computing does not provide services in accordance with Service Level Agreements (SLAs), then it will lose its dependable clients. For the service to satisfy customers, it must meet the requirements set by the QoS parameters.

The issue concerning the power and performance management of virtualised computing environments has been the subject of numerous studies and analyses. Among others by Kusic et al. [14]. The solution to the model of cloud computing that includes energy consumption was devoted to the work by Srikantaiah et al. [20]. An approach to the problem of power-efficient allocation of VMs in virtualised a heterogeneous computing environment was presented by Cardosa et al. [6]. In some of the more recent papers, a. o. [3], an architectural framework in addition to resource allocation principles for energy efficient cloud computing with an allocation algorithm has been proposed. This algorithm takes into account QoS expectations and power usage characteristics of data resources.

In turn, in [8] an optimal minimum energy scheduler for the dynamic online joint allocation of task sizes, computing rates, communication rates and communication powers in virtualised Network Data Centres has been presented. Nevertheless, none of the above studies gave a detailed analysis of the allocation of resources taking into account the values of QoS parameters.

The aim of this research is to present a formal methodology for solving the problem of resource allocation in cloud computing. The developed formalism takes into account both the need to save electricity and guarantee all users the required QoS quality parameters. For effective resolution of this problem, a scheduler has been provided that can be used online for efficient allocation of resources in cloud computing.

The content of the paper is as follows. Section 2 presents the problem of resource allocation in cloud computing. Section 3 contains the proposed solution in the form of a heuristic algorithm. Section 4 presents the results of simulation tests. Section 6 summarizes this paper .

2 System Model

In the adopted cloud computing model, it is assumed that a virtual connection request is required with any virtual machine. The set of servers and set of virtual machines are represented by S and VMS respectively. Let M_{sm} be the amount of resources (e.g. memory) available on a server s, where $s \in S$ and $m \in \{memory(mem), CPU\ unit(cu), storage(sg)\}$. For instance $M_{sm} = 20$ indicates that required memory on server is equal to 20 GB and m denotes a specific type of desired memory. It is assumed that K_{vm} represents the amount of resources needed for every virtual machine v, $v \in VMS$. For instance, $K_{vm} = 14$ denotes that virtual machine v requires 14 GB of memory of m type. The set of network paths and the set of links are represented by P and L, respectively. Let a_{lp} be a binary parameter such that $a_{lp} = 1$ if link $l \in L$ is on path p, $p \in P$, 0 otherwise. For each path is prepared alternate path. An alternative path can be used when, for example, a request is scheduled using a server that is currently busy. For instance, b_{scp} is a binary parameter such that $b_{scp} = 1$ if path $p \in P$ is an alternate path from server s, $s \in S$, to server c, $c \in S$; 0 otherwise. A set of connection requests is defined by I. Every connection request i, $\in I$, is denotes by a source f_i, a destination d_i, start time r_i, connection duration t_i. The allowed accepted delay (tardiness) for each connection request is defined by D_i.

Three binary decision variables are used for a full description of the system: X_{vs}, Y_{ip}, Z_{ij}. The first binary variable, X_{vs}, indicates that virtual machine v is scheduled on server s, $s \in S$. Y_{ip} is a binary decision variable such that Y_{ip} if request i is scheduled on path p. The third binary variable indicates the order of request execution, namely: if $Z_{ij} = 1$, then request i is scheduled before request j.

Power consumption by computing servers in cloud computing depends mainly on consumption by CPU. It is obvious that the greatest energy saving can be achieved by switching idle servers to the sleep mode. It means that the switching

idle servers off to reduce total power consumption. technique of switching idle servers off to reduce total power consumption. It is assumed that the CPU utilization is a function of time and is represented as $u(t)$. Thus, the power consumption model can be defined as [5]:

$$P(u(t)) = k \cdot P_{max} + (1 - k) \cdot P_{max} \cdot u(t) \tag{1}$$

where P_{max} is the maximum power consumed when the server is fully used, k is the fraction of power consumed by the idle server (i.e. 70%).

Quality of service represents functional attributes of the given service based on quantity or quality. Represented services in cloud computing environment have qualitative attributes such as accountability, cost, assurance, privacy, etc. [23]. It is assumed that for the k-th QoS, the attribute at a given sampling interval is formally expressed as

$$QoS_k^q(t) = f(SP_k(t), \delta) \tag{2}$$

where $Q_k^q(t)$ is the discrete or mean value (e.g. average response time) of the k-th QoS attribute from $t-1$ to t. F denotes the QoS function, which change at runtime, δ is the other inputs (e. q. historical timeseries QoS points and tuning variables etc.). $SP_k^q(t)$ indicates the selected primitives matrix of $QoS_k^q(t)$ at t.

It is assumed that all QoS parameters form the so-called collection of possible behaviour primitives spaces. The subset of this set is the selected primitives space, which contains the desired values of QoS parameters for the implementation of this algorithm.

Let A be a series of value over time of QoS attributes that are elements of the possible behavior primitive collection. Let B be a series of value over time of the QoS attribute being part of the desired set. Then you can define a symmetric uncertainty that is equal

$$U(A, B) = \frac{2 \sum_{a \in A} \sum_{b \in B} p(a, b) \log(\frac{p(a,b)}{p(a) \cdot p(b))})}{\sum_{a \in A} p(a) \log(p(a)) + \sum_{b \in B} p(b) \log(p(b))} \tag{3}$$

where a, b are one of the values of A or B, respectively. $p(a, b)$ is the probability between two values, $p(a)$ and $p(b)$ are the marginal probabilities of the values of A or B, respectively. As it has been shown in the paper by [7], for each feature dimension, certain primitives tend to be more relevant to the QoS than others, e.g., the CPU of the underlying virtual machine usually has greater values than the CPU of co-hosted virtual machines.

For each QoS attribute of a given service-instance, the Φ function can be used, which is defined as follows:

$$\Phi(A, b) = \frac{\sum_{a \in A} U(a, b)}{1 + \sum_{a, a' \in A} U(a, a')}, \quad U(a, b) > 0 \tag{4}$$

where a' is also a series of concrete values of a primitive, A indicates the primitives space.

To solve the problem of the allocation of resources in cloud computing, considering the conditions described above, it is possible to use a mixed integer linear programming (MILP) problem [19]. This problem is typically NP-hard. The first objective of the average accepted delay of client connection requests to and from VMS. The average accepted delay is calculated as the difference between the requested start time r_i by the client's connection and the scheduled start time supplied by the scheduler represented by τ_i. For simplicity it is assumed that all clients requests have the same weight for the scheduler.

The second objective of optimization is to maximize the selected QoS attribute, which is reliability, Rel, which is defined as the percentage of requests that being completed less than a threshold (here 30 ms).

The objective functions of the problem are as follows:

$$\max \quad \sum_{i \in I} (\tau_i - r_i)[1 + P(ut)] \tag{5}$$

$$\max_i \Phi(Rel, rel_i) \tag{6}$$

subject to

$$\sum_{v \in VMS} X_{vs} \times K_{vm} \leq M_{sm}. \quad m \in \{m, c, f\} \tag{7}$$

$$\sum_{s \in S} X_{vs} = 1, \quad v \in VMS \tag{8}$$

$$Y_{ip} + (X_{f_i} + X_{d_i} - 4b_{scp}) \leq 2, \quad i \in I, s \in S, c \in S, p \in P \tag{9}$$

$$\sum_{v \in VMS} Y_{ip} = 1, \quad i \in I \tag{10}$$

$$\sum_{p \in P} \{(t_i \times a_{lp} \times Y_{ip}) + (h \times a_{lp} \times Y_{lp}) + (h \times a_{lp} \times Y_{jp})\}$$

$$+ \tau_i - \tau_j + h \times Z_{ij} \leq 5h, \quad i, j, \in I, \quad l \in L \tag{11}$$

$$Z_{ij} + Z_{ji} = 1, \quad i, j \in I \tag{12}$$

$$x_{vs},\ Y_{ip},\ Z_{ij} \in \{0, 1\} \tag{13}$$

$$\tau_i - r_i \geq 0, \quad i \in I \tag{14}$$

$$\tau_i - r_i \leq D_i, \quad i \in I \tag{15}$$

$$\tau_i, r_i \geq 0, \quad i \in I \tag{16}$$

where parameter h is a large number that guarantees that the solution is derived according to the conditions in the given constraint.

Equation (7) ensures that the virtual machine v is allocated to s server with sufficient computing power. Equation (8) guarantees that each virtual machine v will be allocated to one server. Equation (9) ensures that the a connection request i will be assigned to exactly one path. Equation (10) guarantees that a connection request i will be specified for only one alternate path. Equation (11) ensures that at most one connection request i will be scheduled on a certain link at a given time interval. Equation (12) denotes that request i will start before request j. Equations (13) and (14) ensure that the scheduling time will be within the allowed deadline.

3 Heuristic Algorithm for Resource Allocation in Cloud Computing

The optimization model presented in the previous section is very difficult to solve. Therefore, it is desirable to find a sub-optimal solution that can be applied in real time to the allocation of resources in cloud computing. This solution will take into account all the parameters of cloud computing (number of servers, electricity consumption, QoS parameters, etc.). Moreover, for the implementation of this algorithm, it is assumed that it will be take into consideration all main parameters of cloud computing. It is assumed that this algorithm will be a central controller that manages all cloud connection for minimizing electricity consumption and providing QoS parameters to all users.

The problem of resource allocation in a cloud computing is solved using a greedy best-first (GBF) algorithm [17]. While exhaustive search consistently provides the problems' optimal solution, its complexity quickly reduces the practicality of this type of search. The combination of the greedy algorithm and the heuristic, is instrumental in rapidly obtaining near-optimal solutions in to the intractable large solution space. This technique provides to real-time solution of this problem and can be used as Software Defined Network (SDN) controller [13]. SDN controller manages all connection requests with the objective of minimising average allowable deadline and desired reliability.

It has been assumed that the task of allocating resources in the cloud computing is described by state space S. Let Π be the planning task, with operators related to the graph. Searching for this graph is allowed, which is referred to as the action $a_i, a_i \in \{\mathcal{A}\}$. $\{\mathcal{A}\}$ is a finite set of actions. Graph search algorithm using GBF search method is shown in Fig. 2. It always expands nodes with minimal value of heuristic h.

It is assumed here that the heuristic function is calculated only for the newly generated child of the given node w, and for the others it is only modified. The function $h(w)$ represents the cost of the path leading to the node w from the starting node and takes into account all the parameters of this node (reliability, memory, etc.).

Algorithm 1 Graph search algorithm using Greedy Best-First method

```
1: procedure GRAPH_SEARCH(Π: planning task, h: heuristic)
2:     open ← empty set of SearchNodes
3:     closed ← empty set of States
4:     if h(s_0) < ∞ then
5:         open.insert(make_root_node())
6:     end if
7:     while open is not empty do
8:         local ← empty priority queue of SearchNodes ordered by h
9:         n ← fetch_node(open);
10:        local.insert(n);
11:        // perform local GBF search rooted at n
12:        for ∀i ∈ {1, ..., min(1, h(n, state))} do
13:            if local is empty then
14:                break
15:            else
16:                m ← local.pop_min();
17:                closed.insert(m.state)
18:            end if
19:            if m.state ∈ S' then
20:                return extract.path(m)
21:            end if
22:            for ∀⟨a, s'⟩ > with m.state -a-> s' do
23:                if h(s') < ∞ and s' ∉ closed then
24:                    m' ← make_node(m, a, s');
25:                    local.insert(m');
26:                    open.insert_all(local)
27:                end if
28:                return unsolvable
29:            end for
30:        end for
31:    end while
32: end procedure
```

Fig. 2. The outline of used graph search procedure.

4 Simulation Results

To estimate the effectiveness of the proposed method, a data centre simulation consisting of 100 nodes was taken. Each node is modelled to have one CPU core with performance equivalent to 1000 MIPS, 8 GB RAM and 1 TB of storage. It was assumed that the host consumes from 175 W with 0% CPU utilisation, up to 250 W with 100% CPU utilisation.

In order to be able to assess the impact of the algorithm on the consumption of electricity, the first non-power aware policy was used. Next, the proposed heuristic algorithm was used to save electricity through the data centre. The delay was taken into account as a QoS parameter.

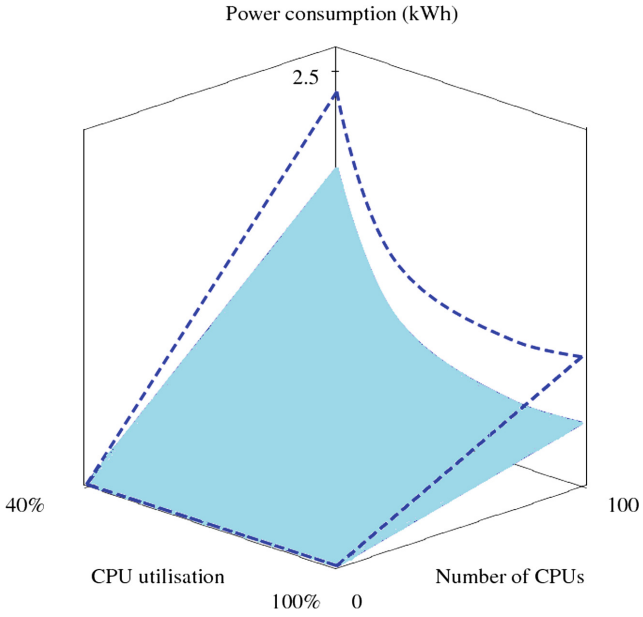


Fig. 3. The power consumption by the cloud. Surface bounded by a dotted line indicates the lack of energy saving.

Figure 3 shows the average energy consumption of the data centre under examination in the absence of energy saving (surface bounded by a dotted line) and in the case of energy saving (surface bounded by a continuous line) for various CPU utilisation rates. It is evident that with the increase of the CPU utilisation, the consumption of electricity is increasing. The impact of the application of the algorithm is evident in particular for a high degree of CPU utilisation. Figure 4 shows the reliability of the cloud computing depending on the number of CPUs used for the system with reliability maximalisation and without it. It is evident that the increasing number of servers increases the reliability of the cloud computing, while the maximization used here improves it significantly by around 15%. Figure 5 shows the percentage number of critical time violations by a virtual machine depending on the processor utilisation for the data centre system with and without the heuristic algorithm used. It is evident that despite the increase in the CPU utilisation rate, the average number of critical time excesses by the virtual machine is much lower for the system with the heuristic algorithm used. The heuristic algorithm used reduces the average number of critical time violations by the virtual machine by up to 20–30%.

One of the important parameters characterizing every cloud computing is virtual machine migrations. Figure 6 shows the number of virtual machine (VMs) migrations versus the mean CPU utilisation for different values of the number of servers. An increase in CPU utilization results in a reduction in the average number of VMs migrations, with a maximum for a given number of servers.

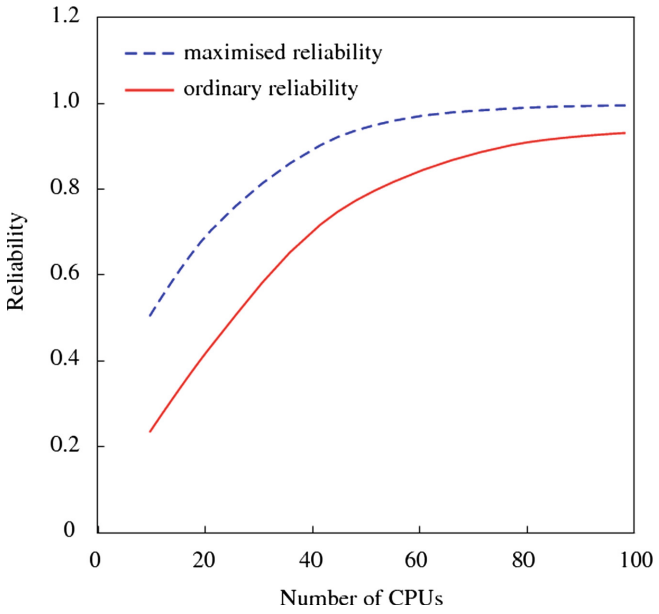


Fig. 4. The reliability of cloud computing.

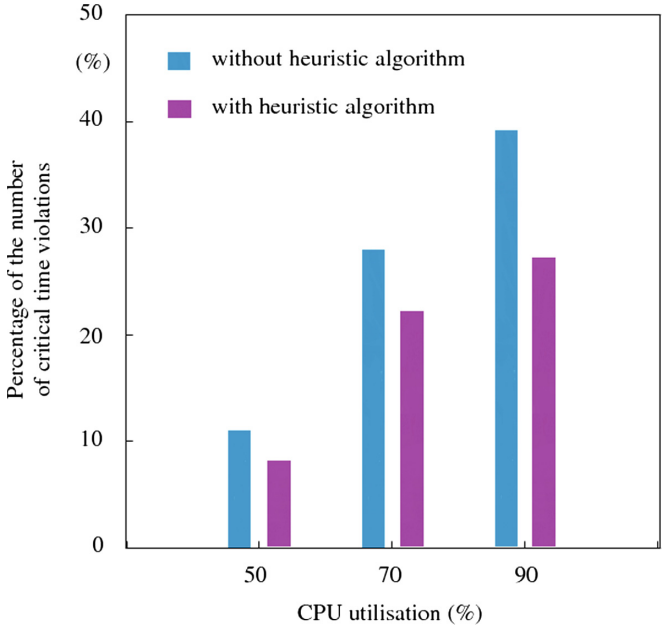


Fig. 5. The percentage number of critical time violations by a virtual machine.

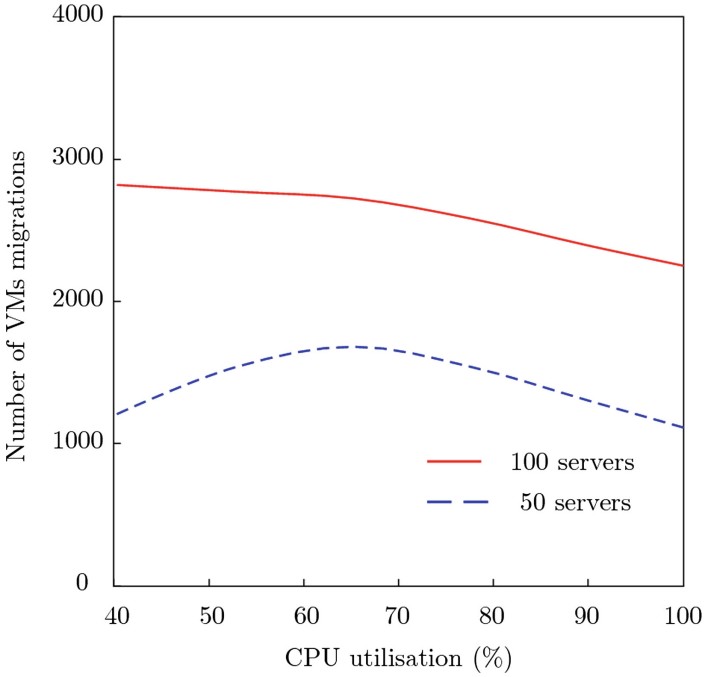


Fig. 6. Number of VMS migration vs. CPU utilisation.

5 Conclusion

This article presents the mathematical framework of the resource allocation in cloud computing. The main purpose of its use is to minimise the consumption of electricity consumed in cloud computing and to meet the requirements of QoS parameters. The proposed heuristic algorithm solves the given problem of allocation of resources in a sub-optimal way, nevertheless the obtained solution is suitable for real-time use. The results of simulation tests confirm the correctness of the obtained solution.

References

1. Amazon Elastic Compute Cloud (EC2). http://www.amazon.com/ec2/. Accessed 10 Jan 2018
2. Amazon Web Services. https://d0.awsstatic.com/whitepapers/aws-overview.pdf. Accessed 12 Jan 2018
3. Beloglazov, A., Abawajy, J., Buyya, R.: Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future Gener. Comput. Syst. **28**(5), 755–768 (2012)
4. Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I.: Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Gener. Comput. Syst. **25**(6), 599–616 (2009)

5. Buyya, R., Beloglazov, A., Abawajy, J.: Energy-efficient management of data center resources for cloud computing: a vision, architectural elements, and open challenges. arXiv preprint arXiv:1006.0308 (2010). arxiv.org
6. Cardosa, M., Korupolu, M.R., Singh, A.: Shares and utilities based power consolidation in virtualized server environments. In: IFIP/IEEE International Symposium on Integrated Network Management. IM 2009, pp. 327–334 (2009)
7. Chen, T., Bahsoon, R.: Self-adaptive and Sensitivity-aware QoS modeling for the cloud, In: The 8th SEAMS, pp. 43–53 (2013)
8. Cordeschi, N., Shojalar, M., Baccarelli, E.: Energy-saving self-configuring networked data center. Comput. Netw. **57**(7), 3479–3491 (2013)
9. Fan, X., Weber, W.D., Barroso, L.A.: Power provisioning for a warehouse-sized computer. In: Proceedings of the 34th Annual International Symposium on Computer Architecture (ISCA 2007), pp. 13–23. ACM, New York (2007)
10. The Green Grid Consortium. http://www.thegreengrid.org
11. Google Cloud Products. https://cloud.google.com/products/. Accessed 10 Jan 2018
12. Hamdaqa, M., Tahvildari, L.: Cloud Computing Uncovered: A Research Landscap, pp. 41–85. Elsevier Press, Amsterdam (2012). ISBN 0-12-396535-7
13. Jammal, M., Taranpreet, S.: A software defined networking: state of the art and research challenges. Comput. Netw. **72**, 74–98 (2014)
14. Kusic, D., Kephart, J.O., Hanson, J.E., Kandasamy, N., Jiang, G.: Power and performance management of virtualized computing environments via lookahead control. Cluster Comput. **12**(1), 1–15 (2009)
15. Millard, C.: Cloud Computing Law. Oxford University Press, Oxford (2013). ISBN 978-0-19-967168-7
16. Ministry of Economy, Trade and Industry, Government of Japan. Establishment of the Japan Data Center Council. Press Release, 4 December 2008
17. Pearl, J.: Heuristics. Addison-Wesley, Reading (1984)
18. Ranganathan, P., Leech, P., Irwin, D., Chase, J.: Ensemble-level power management for dense blade servers. In: ACM SIGARCH Computer Architecture News, vol. 34(2), pp. 66–77. IEEE Computer Society (2006)
19. Rossi, F., Van Beek, P. (eds.): Handbook of Constraint Programming. Elsevier, New York (2006)
20. Srikantaiah, S., Kansal, A., Zhao, F.: Energy aware consolidation for cloud computing. In: Proceedings of the 2008 Conference on Power Aware Computing and Systems, vol. 10, pp. 1–5 (2008)
21. Unnamalai, V.E., Thresphine, J.R.: Service-oriented architecture for cloud computing. Int. J. Comput. Sci. Inf. Technol. **5**(1), 251–255 (2014)
22. Vecchiola, C., Chu, X., Buyya, R.: Aneka: a software platform for .NET-based cloud computing. In: Gentzsch, W., Grandinetti, L., Joubert, G. (eds.) High Speed and Large Scale Scientific Computing. IOS Press, Amsterdam, pp. 267–295 (2009). ISBN: 978-1-60750-073-5
23. Yau, S., Yin, Y.: QoS-based service ranking and selection for service-based systems. In: IEEE International Conference on Services Computing, pp. 56–63 (2011)

Maximum Lifetime of the Wireless Sensor Network and the Gossip Problem

Zbigniew Lipiński(✉)

Institute of Mathematics and Informatics, Opole University, Opole, Poland
zlipinski@math.uni.opole.pl

Abstract. In the gossip problem each node of the graph G possesses a unique piece of information - the gossip message. A sequence of one-way or two-way communications between pair of nodes is made to spread the messages so that any node of the graph knows all the gossips. The question is, what is the minimum number of calls between pairs of nodes needed to exchange all gossip messages? The solution to the two-way communication gossip problem is that $2N-4$ calls ($N \geq 4$) suffice if and only if the graph contains a four cycle subgraph. For one-way communication problem the classical results states that in a strongly connected graph $2N-2$ calls ($N \geq 4$) suffice. In this paper we consider the gossip problem in the context of saving the energy consumed by the sensor network nodes. In the process of gossiping the network nodes consume their energy resources. The amount of consumed energy depends on the cost of transmission between pairs of nodes, a selected transmission route and the sizes of the gossip messages. For the fixed size of gossip messages and the fixed cost of transmission of one unit of data between pairs of nodes we search for the optimal transmission route of the gossip messages such that the most overloaded node consumes the minimum amount of energy. We define the network lifetime for the gossiping process as the time until the first node of the network runs out of its energy. By minimizing the energy of the most overloaded node we obtain the solution to the network lifetime gossiping problem. In the paper we solve the problem for the one-dimensional regular sensor network. We propose a load balancing algorithm for solving the problem in networks with evenly distributed sensors in a given area such that an equal energy solution of the problem exists. We also propose a data aggregation scheduling algorithm to calculate the minimum number of calls necessary to spread the gossip messages for a given solution of the network lifetime gossiping problem.

Keywords: Sensor network lifetime · Gossiping · Energy management

1 Introduction

The gossip problem can be defined in terms of graphs in the following way. In each node of a graph a piece of information called a gossip message is kept. The question is, what is the minimum number of calls between pairs of nodes

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 414–428, 2018.
https://doi.org/10.1007/978-3-319-92459-5_33

needed to exchange all gossip messages? The nodes can exchange the gossip messages by two-way or one-way communication. The original two-way communication gossip problem, also known as the 'telephone problem', was solved in [1–3]. The authors showed that for a complete graph build of $N \geq 4$ nodes to spread all gossip messages $2N - 4$ calls suffice. In [4] it was shown that for arbitrary connected graph $2N - 3$ calls suffice. The 'true conjecture' for the two-way communication gossip problem [4], which states that a connected graph G_N built of N nodes has the solution $2N - 4$ if and only if G_N contains a quadrilateral (a four cycle subgraph), was proven in [5]. Finally, in [6] it was shown that for an arbitrary connected graph G_N, $N \geq 4$, which has no quadrilateral $2N - 3$, two-way calls suffice. The classical result to the one-way gossip problem, also known as the 'telegraph problem', was obtained in [4]. The authors proved that in a strongly connected graph built of N nodes the minimum number of one-direction calls to spread the $N \geq 4$ gossip messages is $2N - 2$. Several generalizations of the problem for various types of graphs and transmission models can be found in [7–9]. Also the complexity aspects of the communication algorithms for the gossip problem have been widely studied. For example studies [10,11] analyzed a class of algorithms with logarithmic complexity which allow one to find the optimal trade-off between the time of gossiping and the number of transmission calls. In [12,13] the authors analyze the averaging problem under the gossip constraint for an arbitrary network graph. A set of averaging algorithms, applicable in random sensor networks, were designed, theirs performance and time complexities were investigated. In [14] two gossip algorithms have been proposed to estimate the number of calls and the total transmission time necessary to broadcast data between two subsets of a given network. The problem of determining the minimum number of rounds necessary to spread all gossips in the network graphs under the restrictions that the nodes can exchange up to fixed number p of messages at each round was considered in [15]. In the case $p = 1$, the authors proposed several optimal algorithms which allow one to solve the problem for a large class of network graphs. An energy efficient gossip algorithm for self-organizing large multi-hop sensor networks with capabilities of the traffic maintenance was proposed by Iwanicki and Steen, [16]. The PL-Gossip algorithm was the first algorithm which allows dynamically manage the multi-level hierarchy structure of the communication network and routing with logarithmic complexity. One of the method of energy saving in wireless sensor networks is the periodical wake up and sleep of the sensors nodes according to some fixed or random time schedule. [17] proposed a design of gossip-based sleep protocols, which allow one to disseminate the management information over the network necessary to put the sensor in the sleep mode. [18] presented the first distributed scheduling gossip algorithm that guarantees maximum throughput in wireless sensor networks. A recent discussion about the application of the gossip problem and the gossip algorithms in communication networks, especially in wireless sensor networks can be found in [11]. A good overview of the gossip problems can be found in [9,19].

In this paper we consider the gossip problem in wireless sensor networks in the context of saving the energy utilized by sensor nodes to extend the network lifetime. We assume that each node of the sensor network S_N is powered by a battery with some initial energy E_0. In the process of dissemination of gossip massages the sensors utilize their energy resources. The amount of energy consumed by each node depends on the graph along which the gossip massages are transmitted and on the size of the gossip messages. We define the network lifetime for the gossiping process as the time until the first node runs out of its energy [20,21]. To extend the network lifetime, i.e., allow the network to repeat the gossiping process maximum number of times, we should minimize the energy consumed by the most overloaded node. Namely, if each node of the network has a battery with the initial energy E_0, then by finding the optimal energy utilization of each node in one cycle of gossiping process E_i^{opt} we can determine the number of cycles $N_{\text{cycles}} = [\frac{E_0}{E_{i'}^{\text{opt}}}]$ the network can perform this process until the most overloaded node i' runs out of its energy. In the presented transmission model the number of calls has no effect on the network lifetime. One can search in the set of solutions to the maximum network lifetime gossiping problem those with the minimum number of calls. We show that in general the minimum number of calls is greater than in the classical gossip problem. We propose a data aggregation scheduling algorithm to determine the minimum number of calls for a given solution. The proposed algorithm aggregates data which can be transmitted simultaneously along the same edge. In the paper we solve the maximum network lifetime gossiping problem for the one dimensional regular sensor network L_N. We show that for the network L_N, in which the nodes are evenly distributed on the line there exists an equal energy solution to the maximum network lifetime gossiping problem. Based on the analysis of this solution we propose an algorithm for solving the problem in networks with evenly distributed sensors in a given area such that an equal energy solution of the problem exists.

2 Definition of the Problem

We define a wireless sensor network as a directed, weighted graph $G = \{S_N, V, E\}$ in which S_N is the set of N network nodes, V is the set of edges, and E set of edges weights. To each edge $t_{i,j} \in V$ we assign a weight $E(t_{i,j}) \equiv E_{i,j}$ defined as the cost of transmission of one unit of data between the i-th and j-th nodes. We assume that the i-th node, $i \in [1, N]$ has a gossip message of size Q_i and this message must be delivered to any other node of the network. The energy consumed by the nodes in the process of data transmission is given by the following formula

$$E_i^s(q) = \sum_j q_{i,j} E_{i,j},$$

where $q_{i,j}$ is the amount of data transmitted between the i-th and j-th nodes in the process of data gossiping. By V_k we denote the set of weighted spanning

trees in G rooted at the k-th node. We can assume, without losing the generality of the problem, that the gossip messages Q_i, $i \in [1, N]$ are transmitted along spanning trees in G. For a given tree $t^{k,r} \in V_k$ rooted at the k-th node we assign the amount of data $q_r^k \geq 0$ transmitted along this tree. It means that along each edge $t_{i,j}^{k,r}$ of the tree $t^{k,r}$ the same amount of data, i.e., $\forall_{t_{i,j}^{k,r} \in t^{k,r}} \, q_{i,j}^{k,r} = q_r^k$, is transmitted. The above assumption allows us to rewrite the formula for the energy consumed by each node $E_i^s(q)$ in terms of trees

$$E_i^s(q) = \sum_{k \in [1,N]} \sum_{t^{k,r} \in V_k} \sum_{j=1, j \neq i,k}^{N} q_r^k \, t_{i,j}^{k,r} \, E_{i,j}, \tag{1}$$

where q is a matrix with elements q_r^k, $k \in [1, N]$, $r \in [1, C_N]$, and the symbol $C_N = |V_k|$ denotes the number of spanning trees in the graph G. The requirement that each node receives the gossip message Q_k can be written in the form

$$\forall_{k \in [1,N]} \sum_r q_r^k = Q_k. \tag{2}$$

Solving the maximum network lifetime gossiping problem, later referred to as *MNLG* problem, means finding a set of trees from $\bigcup_{k \in [1,N]} V_k$ and the amounts of data $\{q_{r_k}^k\}_{k,r_k}$ transmitted along these trees such that the objective function

$$E^s(q) = \max_i \{E_i^s(q)\}_{i=1}^N, \tag{3}$$

reaches its minimum, i.e.,

$$E^s(q^{\text{opt}}) = \min_q E^s(q) = \min_q \max_i \{E_i^s(q)\}_{i=1}^N, \tag{4}$$

where the matrix elements $\forall_{k,r_k} q_{r_k}^k$ of q^{opt} satisfy the requirements (2).

3 Solution to the *MNLG* Problem in L_N

The one-dimensional regular sensor network L_N is built of N nodes placed at the points $x_i = i$ of the line. We assume that $G = \{L_N, V, E\}$ is a complete graph which means that each node of L_N can send data to any other node of the network. The data transmission cost energy matrix $E_{i,j}$ can be any polynomial function of distance $d(x_i, x_j) = |x_i - x_j|$ between transmitter and receiver

$$E_{i,j}(\bar{a}, \bar{\lambda}) = \sum_{n=0}^{\infty} \lambda_n |x_i - x_j|^{a_n}, \tag{5}$$

where $\lambda_n \geq 0$, $a_n \geq 1$. In the L_N network the distance between neighboring nodes is equal to one, $d_{i,i\pm1} = 1$. We can assume, without loosing the generality of the problem (1)–(4) that, $E_{i,i\pm1}(\bar{a}, \bar{\lambda}) = 1$, which is equivalent to the requirement that $\sum_{n=0}^{\infty} \lambda_n = 1$. By E_r we denote the cost of transmission of one unit of data

between nodes which distance is $d_{i,i+r} = r$, $i, r \in [1, N]$. The function E_r defines weights for all edges $t_{i,i+r}$ of a given transmission tree and it has the form

$$E_r(\bar{a}, \bar{\lambda}) = \sum_{n=0}^{\infty} \lambda_n r^{a_n}, \quad r \in [1, N-1], \tag{6}$$

where $\sum_{n=0}^{\infty} \lambda_n = 1$, $\lambda_n \geq 0$ and $a_n \geq 1$.

For fixed values of initial parameters of the *MNLG* problem, i.e., the $E_{i,j}$ matrix and the size of gossip messages Q_i, there is a finite number of solutions of the problem in L_N. We select the permutation invariant one from the set of solutions. Namely, the L_N network is invariant under the following permutation of the nodes $\sigma(i) = N + 1 - i$, $i \in [1, N]$. Also the data transmission cost energy matrix (5) is σ invariant. The invariance follows from the following equation $\sigma(E_{i,j}) = E_{\sigma(i),\sigma(j)} = E_{N+1-i,N+1-j} = E_{i,j}$. If all gossip messages are of the same size, i.e., $\forall_i Q_i = Q_0$, then for a given solution $q_{i,j}$ to the *MNLG* problem the matrix $\sigma(q_{i,j})$ is also a solution. The σ invariance of a solution means that the amount of data transmitted along a given tree $t^{k,r}$ rooted at the k-th node is the same for its mirror tree $\sigma(t^{k,r})$ rooted at $\sigma(k)$ node. The mirror tree $\sigma(t^{i,r})$ is obtained from $t^{k,r}$ by permutation of its edges, i.e., $\sigma(t^{k,r}) = \sigma(t^{k,r}_{i,j}) = t^{\sigma(k),r}_{\sigma(i),\sigma(j)}$. For any solution to the *MNLG* problem given by the set of trees $\{t^{k,r}\}$ and their weights $\{q^k_r\}$ we can construct a symmetric one by permuting $\sigma(t^{k,r})$, $\sigma(q^k_r) \equiv q^{\sigma(k)}_r$ and taking the set of trees $\{t^{k,r}, \sigma(t^{k,r}\}$ with wights $(\frac{1}{2}q^k_r, \frac{1}{2}\sigma(q^k_r))$ as the new solution. For the σ invariant solutions it is enough to consider the optimal behavior for half of the nodes of the network and for the rest of them the solution can be determined from the σ invariance. In this paper we consider the point-to-point transmission model in which the cost of data transmission between pairs of nodes is calculated as the product $q_{i,j}E_{i.j}$. In this model the sensor nodes use the directional antennas for data transmission. In another model, the point-to-multipoint transmission, the nodes use the omnidirectional antennas and the transmission has the wireless multicast advantage property. For the transmission with the wireless multicast advantage property the nodes which are in the range of the transmitting node can receive the data without additional costs of the transmitter. One can show that the obtained solution to the *MNLG* problem in L_N for the point-point transmission model can be easily generalized to the transmission with the wireless multicast advantage property, i.e., the transmission in which the omnidirectional antennas are used. In the following lemma we give a solution to the *MNLG* problem in one-dimensional, regular sensor network L_N with any data transmission cost energy matrix of the form (6).

Lemma. For the directional antennas and arbitrary cost energy matrix $E_{i,j}$ of the form (5), $\forall_{i\in[1,N]}\ Q_i = 1$ the solution to the maximum network lifetime gossiping problem in L_N is given by the following set of trees $\{t^{k,r}\}_{k,r}$ and the amount of data $\{q^k_r\}_{k,r}$ transmitted along these trees

$$
\begin{aligned}
&t^{1,1} = (t_{1,2}, t_{2,3}, \cdots, t_{i,i+1}, \cdots, t_{N-1,N}),\\
&t^{1,2} = (t_{1,2}, \cdots, t_{1,\frac{N}{2}+1}, t_{\frac{N}{2}+1,\frac{N}{2}+2}, \cdots, t_{i,i+1}, \cdots, t_{N-1,N}),\\
&t^{i,1} = (t_{1,2}, t_{2,3}, \cdots, t_{i,i-1}, t_{i,i+1}, \cdots, t_{N-1,N}), \quad i \in [2, N-1],\\
&t^{N,1} = (t_{N,N-1}, t_{N-1,N-3}, \cdots, t_{i+1,i}, \cdots, t_{2,1}),\\
&t^{N,2} = (t_{N,N-1}, \cdots, t_{N,\frac{N}{2}}, t_{\frac{N}{2},\frac{N}{2}-1}, \cdots, t_{i+1,i}, \cdots, t_{2,1}),
\end{aligned}
\tag{7}
$$

$$
q(L_N) = \begin{cases}
q_1^i = Q_i - N\frac{E_1}{\sum_{j=1}^{\frac{N}{2}} E_j}, & i = 1, N,\\
q_2^i = N\frac{E_1}{\sum_{j=1}^{\frac{N}{2}} E_j}, & i = 1, N,\\
q_1^i = Q_i, & i \in [2, N-1],
\end{cases}
\tag{8}
$$

for $N \geq 6$ even, and

$$
\begin{aligned}
&t^{1,1} = (t_{1,2}, t_{2,3}, \cdots, t_{i,i+1}, \cdots, t_{N-1,N}),\\
&t^{1,2} = (t_{1,2}, \cdots, t_{1,\frac{N+1}{2}}, t_{\frac{N+1}{2},\frac{N+1}{2}+1}, \cdots, t_{i,i+1}, \cdots, t_{N-1,N}),\\
&t^{1,3} = (t_{1,2}, \cdots, t_{1,\frac{N+1}{2}+1}, t_{\frac{N+1}{2}+1,\frac{N+1}{2}+2}, \cdots, t_{i,i+1}, \cdots, t_{N-1,N}),\\
&t^{i,1} = (t_{1,2}, t_{2,3}, \cdots, t_{i,i-1}, t_{i,i+1}, \cdots, t_{N-1,N}), \quad i \in [2, N-1],\\
&t^{N,1} = (t_{N,N-1}, t_{N-1,N-2}, \cdots, t_{N-i,N-i-1}, \cdots, t_{2,1}),\\
&t^{N,2} = (t_{N,N-1}, \cdots, t_{N,\frac{N+1}{2}}, t_{\frac{N+1}{2},\frac{N+1}{2}-1}, \cdots, t_{i+1,i}, \cdots, t_{2,1}),\\
&t^{N,3} = (t_{N,N-1}, \cdots, t_{N,\frac{N+1}{2}-1}, t_{\frac{N+1}{2}-1,\frac{N+1}{2}-2}, \cdots, t_{i+1,i}, \cdots, t_{2,1}),
\end{aligned}
\tag{9}
$$

$$
q(L_N) = \begin{cases}
q_1^i = Q_i - 2N\frac{E_1}{E_{\frac{N+1}{2}} + 2\sum_{j=1}^{\frac{N-1}{2}} E_j}, & i = 1, N,\\
q_2^i = N\frac{E_1}{E_{\frac{N+1}{2}} + 2\sum_{j=1}^{\frac{N-1}{2}} E_j}, & i = 1, N,\\
q_3^i = N\frac{E_1}{E_{\frac{N+1}{2}} + 2\sum_{j=1}^{\frac{N-1}{2}} E_j}, & i = 1, N,\\
q_1^i = Q_i & i \in [2, N-1],
\end{cases}
\tag{10}
$$

for $N \geq 3$ odd.

Proof. In one dimension any monomial $E_{i,j} = d(x_i, x_j)^a = |x_i - x_j|^a$, $a \geq 1$ satisfies the inequality

$$
\forall_{0 \leq x_i \leq x_j \leq x_k} \; E(x_i, x_j) + E(x_j, x_k) \leq E(x_i, x_k). \tag{11}
$$

From this property of $E_{i,j}$ follows that the lowest cost of delivery of a given amount of data from the i-th node located at x_i to the j-th node located at x_j is reached, when the data is transmitted between nearest neighbors along the shortest path. By the shortest path we mean the distance between two nodes, i.e., $d(x_i, x_j) = |x_i - x_j|$. From the regularity of the network L_N and the property (11) of the data transmission cost energy matrix (5) follows that the internal nodes of L_N consume the minimal amount of energy when they transmit gossip messages Q_i, $i \in [2, N-1]$ along the trees $t^{i,1}$ from (7) and (9). Because we assumed that the gossip messages are of the same size, i.e., $\forall_{i \in [1,N]} Q_i$, then the energy consumed by each node, given by the formula $E_i^s = E_1(Q_i + \sum_{j=2}^{N-1} Q_j)$, is the same and thus can not be minimized. Let us assume

that the 1-st and the N-th nodes transmit the messages Q_1 and Q_N along the trees $t^{1,1}$ and $t^{N,1}$ from (7) and (9), then the energy consumed by the internal nodes equals to $E_i^s = E_1(Q_i + \sum_{j=1}^{N} Q_j)$. For $Q_i = 1$ this formula has the form $\forall_{i\in[2,N-1]}\ E_i^s = (N+1)E_1$ and $E_1^s = 1$, $E_N^s = 1$. Transmission of all of the gossip messages along the set of trees $\{t^{i,1}, t^{1,1}, t^{N,1}\}_{i\in[2,N-1]}$ is not optimal because the border nodes (the 1-st and the N-th) have unspent energy. To balance the energy consumption we should decrease the energy consumed by the internal nodes of L_N, i.e., $\forall_{i\in[2,N-1]}\ E_i^s$ and increase the energy consumed by the 1-st and the N-th node. The optimal transmission strategy for the 1-st node is to send the maximum amount of data to the $(\frac{N}{2}+1)$-th node for N even and to the $\frac{N+1}{2}$-th and $(\frac{N+1}{2}+1)$-th nodes for N odd, and the rest of data along the tree $t^{1,1}$ such that the energy consumed by all nodes of the network is equal. Through the use of permutation operations σ we obtain the optimal transmission tree for the N-th node ($\sigma(1) = N$). The energy balance between internal and border nodes of L_N network is possible if

$$E_1^s \geq NE_1, \tag{12}$$

where E_1 denotes the energy defined by the formula (6) for $r = 1$. We show that the inequality (12) is satisfied when at least one of the exponents a_n in (5) is greater or equal to one. This is also the sufficient condition for the trees (7) and (9) to be the solution to the *MNLG* problem in L_N. Let us assume that the gossip message $Q_1 = 1$ is split into several pieces q_m^1, i.e., $\sum_m q_m^1 = Q_1$, and it is transmitted by the 1-st node along the set of edges $\{t_{1,j}^{1,r}\}_{j\in[2,m]}$, where $m = \frac{N}{2}+1$ for N even and $m = \frac{N+1}{2}+1$ for N odd. For such transmission the energy of the first node is given by the expression

$$E_1^s = \sum_{j=1}^{m}\sum_{r} q_m^1 t_{1,j}^{1,r} E_{1,j}.$$

To balance the energy between all nodes of the L_N network the inequality (12) must be satisfied. For $E_{i,j} = |i-j|^a$ it has the form $\sum_{j=1}^{K} j^a \geq 2K$ for $N = 2K$ and $\sum_{j=1}^{K+1} j^a \geq (2K+1)$ for $N = 2K+1$. For $a = 1$ it is trivially satisfied for $N \geq 6$, N even and $N \geq 3$, N odd. From this follows that the inequality (12) is also satisfied for any cost energy matrix $E_{i,j} = |i-j|^a$, $a \geq 1$ and more generally for (5). ◇

The optimal transmission trees for the 1-st node are shown in Figs. 1 and 2. Because of the σ invariance of L_N and $E_{i,j}$ after exchanging $\sigma(i) = N+1-i$ in L_N we get the optimal trees for the N-th node.

4 The Minimum Node Weight Spanning Trees

In the set of trees (7) and (9) that form the solution to the *MNLG* problem in L_N network there are special types of trees $t^{i,1}$, $i \in [1, N]$. We will call them the

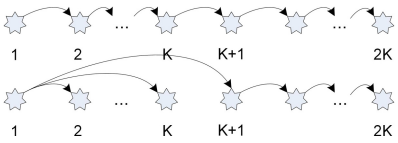


Fig. 1. Optimal transmission trees for the first node in L_{2K}.

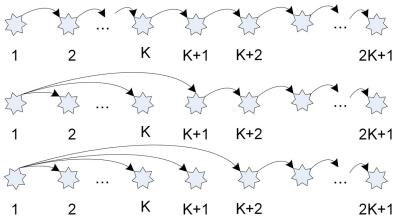


Fig. 2. Optimal transmission trees for the first node in L_{2K+1}.

minimum node weight spanning trees in L_N. This type of trees was introduced in [22,23] in the context of solving the maximum network lifetime broadcasting problem in sensor networks. We will show on the examples that such trees play also an important role in solutions to the *MNLG* problem in sensor networks. The definition of the minimum node weight spanning tree requires introduction of the in- and out-weight of a graph's node. In our transmission model the directed edge $t_{i,j}$ represents the communication link between the transmitter and the receiver. The directed edge $t_{i,j}$ is the out-edge of the i-th node and the in-edge of the j-th node. To each directed edge $t_{i,j}$ we assigned the data transmission cost energy matrix $E_{i,j}$ as its weight. We define the out-weight of the i-th node in the subgraph $G' \subseteq G$ as the sum of weights of the out-edges incident with the node, i.e.,

$$E_i(G') = \sum_{t_{i,j} \in V(G')} E_{i,j}.$$

We write the out-weights of all nodes in G' as the vector

$$\bar{E}(G') = (E_1(G'), \cdots, E_N(G')) = \sum_{t_{i,j} \in V(G')} \bar{E}_{i,j},$$

where $\bar{E}_{i,j} = (0, \cdots, E_{i,j}, \cdots, 0)$. In a similar way the in-weight of the node can be defined. In the transmission model considered in this paper we assume that there is no cost of data receiving and thus the in-weight of the graph nodes are equal to zero. The minimum node weight spanning tree $t^{MNW} \in V_k$ rooted at the k-th node is a tree which node with the maximum out-weight is minimal among spanning trees in V_k, i.e,

$$\forall_{t \in V_k} \max\{E_i(t^{\mathrm{MNW}})\}_{i=1}^{N} \leq \max\{E_i(t)\}_{i=1}^{N}.$$

Let us consider as an example the maximum network lifetime gossiping problem in a two-dimensional sensor network S_5 built of five nodes. We assume that the

nodes are located at the points of the plane $x_1 = \{0, 2\}$, $x_2 = \{2, 2\}$, $x_3 = \{3, 0\}$, $x_4 = \{4, 3\}$, $x_5 = \{5, 2\}$ and the out-weights of the edges are given by the data transmission cost energy matrix $E_{i,j} = d(x_i, x_j)^4$. In S_5 there are six directed minimum node weight spanning trees. Four of them are rooted in the nodes 1, 3, 4, 5 and are constructed from the minimum spanning tree in S_5. The two minimum node weight spanning trees rooted at the 2-nd node are $t_{2.1}^{\mathrm{MNW}} = (t_{2,1}, t_{2,4}, t_{4,5}, t_{5,3})$, $t_{2.2}^{\mathrm{MNW}} = (t_{2,1}, t_{2,3}, t_{3,5}, t_{5,4})$. The Table 1 contains a set of trees and the corresponding amounts of data transmitted along these trees which form a solution to the *MNLG* problem in S_5 for $Q_i = 1$, $i \in [1, 5]$. In Table 1 there are two minimum node weight spanning trees. The $t^{2,1}$ tree rooted at the 2-nd node and $t^{5,1}$ tree rooted at the 5-th node. For the solution each node consumes the same amount of energy equal to $\forall_i E_i = 122.41$.

Table 1. Set of trees which solve the *MNLG* problem in S_5.

Tree	Data
$t^{1,1} = (t_{1,2}, t_{1,3}, t_{2,4}, t_{4,5})$	$q_1^1 = 0.16$
$t^{1,2} = (t_{1,2}, t_{2,4}, t_{4,3}, t_{4,5})$	$q_2^1 = 0.39$
$t^{1,3} = (t_{1,2}, t_{2,3}, t_{3,5}, t_{5,4})$	$q_3^1 = 0.45$
$t^{2,1} = (t_{2,1}, t_{2,4}, t_{4,5}, t_{5,3})$	$q_1^2 = 0.53$
$t^{2.2} = (t_{2,1}, t_{1,3}, t_{3,5}, t_{5,4})$	$q_2^2 = 0.47$
$t^{3,1} = (t_{3,5}, t_{5,4}, t_{4,2}, t_{2,1})$	$q_1^3 = 1.00$
$t^{4,1} = (t_{4,2}, t_{2,1}, t_{4,5}, t_{5,3})$	$q_1^4 = 1.00$
$t^{5,1} = (t_{5,4}, t_{4,2}, t_{2,1}, t_{2,3})$	$q_1^5 = 0.80$
$t^{5,2} = (t_{5,4}, t_{4,2}, t_{2,1}, t_{5,3})$	$q_2^5 = 0.20$

In the next Section we propose an algorithm for solving the maximum network lifetime gossiping problem in sensor networks in which an equal energy solution exists. By means of the algorithm one can find an approximate solution to the *MNLG* problem by balancing the load of network nodes when the gossip data is transmitted along the minimum node weight spanning trees in a given network. To find such trees in a network graph one can use the minimum node weight spanning tree searching algorithms discussed in [22,23].

4.1 The Load Balancing Gossip Algorithm

The purpose of the load balancing gossip algorithm (the *LBG* algorithm) is to find the set of trees V^{LBG} which allow one to split the data transmitted along a given set of minimum node weight spanning trees V^{MNW}in S_N in such a way that the energy consumed by the nodes of the network is more evenly distributed. In the first iteration of the algorithm which determines the approximate solution of the *MNLG* problem we calculate the minimum of the objective function (3)

over the set V^{MNW} of all minimum node weight spanning trees in S_N. In the second iteration by means of the *LBG* algorithm we construct the set of trees V^{LBG} which allow to balance the energy consumed by network nodes when the gossip messages are transmitted along the minimum node weight spanning trees in S_N. To state whether the energy of a node must be decreased or increased, we define the average energy consumed by network nodes for a given minimum node weight spanning tree t and the whole set V^{MNW}. If we assume, that the k-th node sends a gossip message of the size Q_k along some tree $t^{k,r}$, then we calculate the average energy utilized by nodes of the S_N network by means of the following formula

$$\bar{E}(t^{k,r}) = Q_k \frac{\sum_{i\in[1,N]} E_i(t^{k,r})}{N'_{t^{k,r}}}, \tag{13}$$

where $N'_{t^{k,r}}$ is the number of nodes in the tree $t^{k,r}$ which transmit data, i.e., we do not count the terminal nodes. For nodes which consume more energy that the average $\bar{E}(t) \leq E_i(t)$ we will search for trees which allow us to decrease their energy consumption. For nodes which consume less energy than the average $E_i(t) < \bar{E}(t)$ we will increase their energy consumption. We define the second criterion which allows us to state whether the energy consumed by the node must be decreased or increased. We introduce the average energy consumed by the sensor nodes transmitting the gossip messages over all trees from the set V^{MNW}

$$\bar{E}(V^{\mathrm{MNW}}) = \frac{1}{|V^{\mathrm{MNW}}|} \sum_{k,r} \sum_{t^{k,r}\in V^{\mathrm{MNW}}} \bar{E}(t^{k,r}). \tag{14}$$

We compare the average energy $\bar{E}(V^{\mathrm{MNW}})$ with

$$E_i(V^{\mathrm{MNW}}) = \frac{1}{|V^{\mathrm{MNW}}|} \sum_{k,r} \sum_{t^{k,r}\in V^{\mathrm{MNW}}} E_i(t^{k,r}).$$

This criterion can be applied when the minimum node weight spanning tree $t^{k,r}$ rooted at the k-th node is used to balance the energy consumed by nodes transmitting the gossip message Q_i for $i \neq k$.

By $V_i^e(t) = \{t_{i,j}\}_j$ we denote the set of outgoing edges of the i-th node in a given tree $t \in V^{\mathrm{MNW}}$. The following procedure is applied in the *LBG* algorithm to decrease the energy consumed in the node.

Procedure 1. Spanning tree graph construction for decreasing energy consumed in the node.

To decrease the energy consumed by the i-th node remove one outgoing edge $t_{i,r}$ from the set $V_i^e(t)$ and construct the minimum node weight spanning tree t' rooted at the i-th node for the graph (S_N, V', E), where $V' = V \setminus \{t_{i,r}\}$, such that the inequality

$$E_i(t') < E_i(t) \tag{15}$$

is satisfied. Repeat the process of constructing the minimum node weight spanning tree in S_N with removed one, two and finally all edges from a given non-root node. For the root node leave at least one outgoing edge. ◇

The requirement (15) allows us not only to remove edges from the node but also to add to the i-th node a set of edges $\{t_{i,s}\}_s$ satisfying the inequality $\sum_s t_{i,s}E_{i,s} < t_{i,r}E_{i,r}$. If the constructed tree t' is not unique, we can select one tree from the equivalent trees or add all equivalent trees to the set V^{LBG}.

The second procedure is applied in the *LBG* algorithm to increase the energy consumed by nodes of the network.

Procedure 2. Spanning tree graph construction for increasing energy consumed in the node.
To increase the energy consumed by the i-th node add to the set of outgoing edges $V_i^e(t)$ a new edge $t_{i,r}$ and construct the minimum node weight spanning tree t' in S_N rooted at the i-th node, such that

$$V_i^e(t') = V_i^e(t) \bigcup \{t_{i,r}\}. \tag{16}$$

Repeat the process of constructing the minimum node weight spanning trees t' with added one, two and finally all $(N-2)$ edges to the non-root node and $(N-1)$ edges to the root node. ⋄

The Load Balancing Gossip Algorithm (the *LBG* algorithm).
Step 1. *Select a minimum node weight spanning tree t from the set V^{MNW}. By means of the formula (13) determine the average energy $\bar{E}(t)$ utilized by nodes of the network S_N for the tree t.*
Step 2. *For nodes which energy is greater than the average $\bar{E}(t) \leq E_i(t)$ apply the Procedure 1, i.e. construct the set of trees for decreasing the energy consumed by these nodes V^{LBG}. For nodes which the energy consumed along the tree t is less than the average $E_i(t) < \bar{E}(t)$ apply the Procedure 2, construct the set of trees for increasing the energy consumed by these nodes.* ⋄

In the above algorithm we can use the average energy utilization criterion (14). For this purpose, in the Step 2 of the *LBG* algorithm instead of the inequality $\bar{E}(t) \leq E_i(t)$ the inequality $\bar{E}(V^{\mathrm{MNW}}) < E_i(V^{\mathrm{MNW}})$ should be used.

For each minimum node weight spanning tree t in S_N we are able to construct by means of the *LBG* algorithm a set of trees V^{LBG} which allow us to balance the energy consumed by nodes of the network in the process of data gossiping. In the final stage of solving the *MNLG* problem by means of the proposed algorithms we define the objective function (3) over the set of trees V^{LBG} and determine its minimum with respect to the set of variables q_r^k, $t^{k.r} \in V^{\mathrm{LBG}}$.

We apply the algorithm to find the solution to the *MNLG* problem in S_5 network. According to the second criterion (14), the average energy consumed be sensor nodes is equal to $E(V^{\mathrm{MNW}}) = (2.67, 42.5, 14.83, 11.0, 12.0)$ and $\bar{E}(V^{\mathrm{MNW}}) = 27.67$. This means that we should find a set of trees which allows us to decrease the energy of the 2-nd node and increase the energy consumed by the rest of the nodes in S_5. The following table shows the application of the *LBG* algorithm to solve the *MNLG* problem in S_5. To decrease the energy consumed by the 2-nd node we took the minimum node weight spanning tree t_1^{MNW} rooted at the 1-st node. By shifting edges of the t_1^{MNW} tree according to Procedure 1

Table 2. Application of the load balancing algorithm to V^{MNW} in S_5.

Tree	Subtree replacement	New tree	Increase	Decrease
			E_i^s	E_i^s
t_1^{MNW}	$t_{2,3}$ by $t_{1,3}$	$t^{1,1}$	1	2
t_1^{MNW}	$t_{2,3}$ by $t_{4,3}$	$t^{1,2}$	4	2
t_1^{MNW}	$(t_{2,4}, t_{4,5})$ by $(t_{3,5}, t_{5,4})$	$t^{1,3}$	3, 5	2
$t_{2.2}^{\mathrm{MNW}}$	$t_{2,3}$ by $t_{1,3}$	$t^{2,2}$	1	2
t_3^{MNW}	$(t_{3,2}, t_{2,4}, t_{4,5})$ by $(t_{3,5}, t_{5,4}, t_{4,2})$	$t^{3,1}$	3, 4, 5	2
t_4^{MNW}	$t_{2,3}$ by $t_{5,3}$	$t^{4,1}$	5	2
t_5^{MNW}	$t_{2,3}$ by $t_{5,3}$	$t^{5,2}$	5	2

and Procedure 2, shown in the first three rows of the Table 2, we obtained tress $t^{1,r}$, $r = 1, 2, 3$, which belong to the solution set of the *MNLG* problem in S_5. Another four trees from the solution were obtained by applying the shifts of the edges in the minimum node weight spanning trees $t_{2.2}^{\mathrm{MNW}}$, t_3^{MNW}, t_4^{MNW}, t_5^{MNW}. The two missing trees in Table 2 which form the solution of the *MNLG* problem in S_5 are in the set V^{MNW}. These are $t^{2,1}$ and $t^{5,1} = (t_{5,4}, t_{4,2}, t_{2,1}, t_{2,3})$ trees. In the above example, by applying the load balancing gossip algorithm we obtained all trees which form the solution to the *MNLG* problem in S_5.

5 Scheduling Algorithm for the *MNLG* Problem

The definition of the maximum network lifetime gossiping problem (1)–(4) does not impose any conditions on the number of calls the gossips messages are spread over the network. We can assume that the gossip messages are sent sequentially over the network along each edge of the transmission trees. If in the solution of the *MNLG* problem there are N' trees, then in the case of the sequential transmission $(N-1)N'$ calls must be performed to spread all the gossips. For example, in L_N network the number of sequential calls equals to $(N-1)(N+2)$ for N even and $(N-1)(N+4)$ for N odd. In this Section we propose an algorithm which allows one to find the minimum number of calls necessary to spread all gossip messages for a given solution of the *MNLG* problem. To minimize the number of calls we aggregate the gossip data q_r^k transmitted along the same edge of the trees $t^{k,r}$ so that the parallel transmission can be performed. To do it, we represent the transmission of gossip data along a given tree as transmissions along set of paths in the tree. This allows us to organize the transmission in rounds. In each transmission round we can aggregate the gossip data transmitted along the same edge in a one call. We denote the set of all paths for a given solution of the *MNLG* problem by

$$V^P = \{p^{k,r,s}\}_{k,r,s},$$

the index s numbers the paths in the tree $t^{k,r}$. We define the "history path" $p_{i,j}^{H(k,r,s)}$ of the edge $t_{i,j}^{k,r,s}$ in $p^{k,r,s}$ as the sequence of all edges in $p^{k,r,s}$ which precede the given edge. The "future path" $p_{i,j}^{F(k,r,s)}$ of the edge $t_{i,j}^{k,r,s}$ is the sequence of all edges in $p^{k,r,s}$ which follow the given edge. We denote the set of edges in the paths $p_{i,j}^{H(k,r,s)}$ and $p_{i,j}^{F(k,r,s)}$ by $V_{i,j}^{H(k,r,s)}$ and $V_{i,j}^{F(k,r,s)}$, respectively. The aggregation of the data q_r^k and $q_{r'}^{k'}$ transmitted along the edges $t_{i,j}^{k,r}$ and $t_{i,j}^{k',r'}$ means summing up the data $(q_r^k + q_{r'}^{k'})$ and transmitting it along one of the edge $t_{i,j}^{k,r}$ or $t_{i,j}^{k',r'}$. In general, the data q_r^k is not transmitted along all edges of the set $\{p^{k,r,s}\}_s$, where k and r are fixed. The data q_r^k is transmitted only along one edge of the overlapping edges $t_{i,j}^{k,r,s} = t_{i,j}^{k,r,s'}$ in $p^{k,r,s}$. To avoid this ambiguity, we mark the overlapping edges $t_{i,j}^{k,r,s} = t_{i,j}^{k,r,s'}$ in $p^{k,r,s}$ along which the data q_r^k is not transmitted. A transmission round R_α is a set of edges $t_{i,j}^{k,r} \in V$, for fixed i and j, along which different data can be transmitted in parallel. Aggregation of a data can be performed only when the edges belong to the same transmission round. Two edges can be put to the same transmission round by shifting the edges and their future paths. Shifting the edge $t_{i,j}^{k,r}$ from the round R_α to the round $R_{\alpha'}$ must keep the order of the edges in the path $p^{k,r,s}$ to which the edge $t_{i,j}^{k,r}$ belongs. In the set of paths V^P we find the paths which satisfy the condition

$$\min_{i,j} |(\bigcup_{k,r,s} V_{i,j}^{H(k,r,s)}) \bigcap (\bigcup_{k,r,s} V_{i,j}^{F(k,r,s)})| \tag{17}$$

If the set (17) is empty, for fixed i and j, it means that all edges of the set $V_{i,j}^{\alpha} = \{t_{i,j}^{k,r,s}\}_{k,r,s}$ can be put to the same transmission round. We place the edges from the set V^P in the initial round set $V_0^R = (R_1, \cdots, R_M)$, where M is the length of the longest path in V^P, i.e., $M = \max_{k,r,s} |p^{k,r,s}|$, where $|p^{k,r,s}|$ denotes the number of edges in $p^{k,r,s}$.

Let $V^{\text{sol}} = \bigcup_k V_k^{\text{sol}}$ denotes the set of tress which form the solution to the *MNLG* problem, where V_k^{sol} is a set of solution trees rooted at the k-th node.

The aggregation scheduling algorithm for the *MNLG* problem.

Step 1. *For each tree $t^{k,r}$ from the set V^{sol} extract all distinct directed paths $p^{k,r,s}$. Mark the overlapping edges $t_{i,j}^{k,r,s} = t_{i,j}^{k,r,s'}$ in the set $\{p^{k,r,s}\}_s$ along which the data q_r^k is not transmitted.*

Step 2. *For each edge $t_{i,j}^{k,r,s}$ from V^P determine the "history path" $P_{i,j}^{H(k,r,s)}$ and the "future path" $P_{i,j}^{F(k,r,s)}$. For fixed i and j select the set of edges $V_{i,j}^{\alpha}$ so that the condition (17) is satisfied. From among the sets $V_{i,j}^{\alpha}$, $i, j \in [1, N]$ select the one with the maximum number of elements, i.e, the set for which $\max_\alpha |V_{i,j}^{\alpha}|$, and put the edges from $V_{i,j}^{\alpha}$ to one transmission round R_α.*

Step 3. *Repeat the step 2 for sub-paths on both sides of the fixed round R_α. The set with the maximum number of the same edges put to the transmission round R_i, where i is the farthest location of the edge in the paths.*

Step 4. *Minimize the number of transmission rounds by aggregation independent calls of the neighboring rounds.* ◇

We will present the application of the aggregation scheduling algorithm on the example of a solution to the *MNLG* problem in S_5 network, see Table 1. In Table 3 the gossip data q_r^k satisfies the equations $q_1^1+q_2^1+q_3^1 = Q_1$, $q_1^2+q_2^2 = Q_2$, $q_1^5+q_2^5 = Q_5$. The total number of edges in $V^{\mathrm{sol}}(S_5)$ is 36. For the sequential transmission of gossip data it is necessary to perform 36 sequential calls in 36 transmission rounds. The aggregation scheduling algorithm allows one to find transmission method in which there are 13 calls grouped in 6 transmission rounds.

Table 3. Solution to the *MNLG* problem in S_5 with scheduling.

Data\Round	1	2	3	4	5	6
q_1^1	$t_{1,2}$	$t_{2,4}$, $t_{1,3}$	$t_{4,5}$	–	–	–
q_2^1	$t_{1,2}$	$t_{2,4}$	$t_{4,3}$, $t_{4,5}$	–	–	–
q_3^1	$t_{1,2}$	$t_{2,3}$	$t_{3,5}$	$t_{5,4}$	–	–
q_1^2	–	$t_{2,4}$	$t_{4,5}$	$t_{5,3}$	–	$t_{2,1}$
q_2^2	$t_{2,1}$	$t_{1,3}$	$t_{3,5}$	$t_{5,4}$	–	–
Q_3	–	–	$t_{3,5}$	$t_{5,4}$	$t_{4,2}$	$t_{2,1}$
Q_4	–	–	$t_{4,5}$	$t_{5,3}$	$t_{4,2}$	$t_{2,1}$
q_1^5	–	–	–	$t_{5,4}$	$t_{4,2}$	$t_{2,1}$,$t_{2,3}$
q_2^5	–	–	–	$t_{5,3}$,$t_{5,4}$	$t_{4,2}$	$t_{2,1}$
#Calls	2	3	3	2	1	2

6 Conclusions

In the paper we considered the equal energy solutions of the maximum network lifetime gossiping problem in sensor networks. We solved the problem in one-dimensional regular sensor network L_N. Based on the analysis of the solution in L_N we proposed the load balancing gossip algorithm for solving the problem in networks in which an equal energy solution of the problem exists. We also proposed the data aggregation scheduling algorithm which allows one to calculate the minimum number of calls necessary to spread gossip messages for a given solution of the maximum network lifetime gossiping problem. We showed the efficiency of the proposed algorithms on the example. For future work we leave the problem of estimating the minimum number of calls necessary to spread the gossip messages for any solution of the maximum network lifetime gossiping problem. Also a detailed study requires the proposed aggregation scheduling algorithm and development of new scheduling algorithms for the maximum network lifetime gossiping problem.

References

1. Tijdeman, R.: On a telephone problem. Nieuw Arch. Wisk. **19**, 188–192 (1971)
2. Hajnal, A., Milner, E.C., Szemeredi, E.: A cure for the telephone disease. Can. Math. Bull. **15**, 447–450 (1972)
3. Baker, B., Shostak, R.: Gossips and telephones. Discrete Math. **2**, 191–193 (1972)
4. Harary, F., Schwenk, A.J.: The communication problem on graphs and digraphs. J. Franklin Inst. **297**, 491–495 (1974)
5. Bumby, R.: A problem with telephones. SIAM J. Algebraic Discrete Methods **2**, 13–19 (1981)
6. Labahn, R.: The telephone problem for trees. Elektron. Informationsverarb. Kybernet. **22**, 475–485 (1986)
7. West, D.: A class of solutions to the gossip problem part I. Discrete Math. **39**, 307–326 (1982)
8. Flouri, K., Beferull-Lozano, B., Tsakalides, P.: Optimal gossip algorithm for distributed consensus SVM training in wireless sensor networks. In: 16th International Conference on Digital Signal Processing, Santorini-Hellas, Greece (2009)
9. Shah, D.: Gossip algorithms. Found. Trends Netw. **3**(1), 1–125 (2008)
10. Czumaj, A., Gasieniec, L., Pelc, A.: Time and cost trade-offs in gossiping. SIAM J. Discrete Math. **11**(3), 400–413 (1998)
11. Hromkovic, J., Klasing, R., Pelc, A., Ruzicka, P., Unger, W.: Dissemination of Information in Communication Networks. Springer, Heidelberg (2005). https://doi.org/10.1007/b137871
12. Boyd, S., Ghosh, A., Prabhakar, B., Shah, D.: Gossip algorithms: design, analysis, and applications. In: Proceedings of the IEEE INFOCOM, Miami, FL (2005)
13. Boyd, S., Ghosh, A., Prabhakar, B., Shah, D.: Randomized Gossip algorithms. IEEE Trans. Inf. Theory **52**(6), 2508–2530 (2006)
14. Lee, H.-M., Chang, G.J.: Set to set broadcasting in communication networks. Discrete Appl. Math. **40**, 411–421 (1992)
15. Bermond, J.-C., Gargano, L., Rescigno, A.A., Vaccaro, U.: Fast gossiping by short messages. SIAM J. Comput. **27**, 917–941 (1998)
16. Iwanicki, K. van Steen, M.: The PL-Gossip Algorithm, Technical report IR-CS-034 Vrije Universiteit Amsterdam (2007)
17. Hou, X., Tipper, D., Wu, S.: A Gossip-based energy conservation protocol for wireless ad hoc and sensor networks. J. Netw. Syst. Manag. **14**(3), 381–414 (2006)
18. Modiano, E., Shah, D., Zussman, G.: Maximizing throughput in wireless networks via gossiping. In: Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems SIGMETRICS, France, pp. 27–38 (2006)
19. Hedetniemi, S.M., Hedetniemi, S.T., Liestman, A.L.: A survey of gossiping and broadcasting in communication networks. Networks **18**, 319–349 (1988)
20. Chang, J.H., Tassiulas, L.: Energy conserving routing in wireless ad-hoc networks. In: Proceedings INFOCOM, pp. 22–31 (2000)
21. Acharya, T., Paul, G.: Maximum lifetime broadcast communications in cooperative multihop wireless ad hoc networks: centralized and distributed approaches. Ad Hoc Netw. **11**, 1667–1682 (2013)
22. Lipiński, Z.: Maximum lifetime broadcasting problem in sensor networks (2015). http://arxiv.org/abs/1511.05587. Wireless Personal Communications (2018, to appear)
23. Lipiński, Z.: Minimum node weight spanning trees searching algorithm for broadcast transmission in sensor networks. In: Proceedings of the 12th International Conference on Digital Information Management (ICDIM 2017), Fukuoka, Japan (2017)

Wireless Network with Bluetooth Low Energy Beacons for Vehicle Detection and Classification

Marcin Bernas[1(✉)], Bartłomiej Płaczek[2], and Wojciech Korski[3]

[1] Faculty of Mechanical Engineering and Computer Science, University of Bielsko-Biala, Bielsko-Biala, Poland
marcin.bernas@gmail.com
[2] Institute of Computer Science, University of Silesia, Katowice, Poland
placzek.bartlomiej@gmail.com
[3] Institute of Innovative Technologies EMAG, Katowice, Poland
wojciech.korski@ibemag.pl

Abstract. The paper proposes a hybrid wireless network, which can be installed on a roadside to detect passing vehicles and recognize their classes. The vehicle detection and classification tasks are performed by analyzing strength of a radio signal received from Bluetooth Low Energy beacons with the use of machine learning algorithms. The introduced system is cost-efficient, easy to install, and can be used for a long time without an external power source. An energy-aware algorithm is proposed, which uses a scheduling mechanism to manage wireless nodes that can act as BLE beacons (in low energy mode) or receivers. Results of experimental evaluation confirm that the proposed solution enables collection of accurate traffic data in real time and prolongs lifetime of battery-powered wireless nodes in the traffic monitoring system. The paper also discusses the applicability of various wireless communication technologies and the influence of wireless node location on vehicle detection accuracy.

Keywords: Wireless network · RSSI · BLE · iBeacon
Machine learning · Data mining

1 Introduction

Road traffic monitoring aims at collecting data that are necessary for optimization of traffic control and management as well as predicting future transportation needs and establishing development plans [1,2]. Key functionalities of traffic monitoring include detection and classification of vehicles at selected locations. The information about presence of vehicles and their classes is utilized, for instance, by adaptive traffic signal systems that are designed to reduce travel time, congestion, and emissions [3].

The most popular solutions for vehicle detection and classification are based on dedicated sensors, e.g., inductive loops, video-detectors, and magnetometers [4].

P. Gaj et al. (Eds.): CN 2018, CCIS 860, pp. 429–444, 2018.
https://doi.org/10.1007/978-3-319-92459-5_34

A drawback of these solutions is a high installation and maintenance cost. In this paper feasibility of an alternative approach to vehicles detection and classification is analyzed. The main idea behind the analyzed approach is to use of-the-shelf wireless communication devices and detect disturbances in received radio signal that are caused by passing vehicles.

According to the proposed traffic monitoring approach, a wireless network is composed of battery-powered Bluetooth Low Energy (BLE) devices. The BLE devices are installed at different heights on both sides of a road. The devices placed on one side of a road use the iBeacon protocol to broadcast their identifiers, while the devices on the opposite side act as receivers and collect values of the received signal strength indicator (RSSI). The RSSI measurements are sent via cellular network to a server, which detects vehicles and categorizes them into classes by analyzing variation of the collected RSSI values. These tasks are performed with the use of machine learning algorithms. The introduced system is cost-efficient, easy to install, and can be used for a long time without additional power source.

The paper is organized as follows. Section 2 includes a survey of related literature and describes contribution of this work. Details of the proposed traffic monitoring system are discussed in Sect. 3. Experiments and their results are described in Sect. 4. Finally, conclusions are given in Sect. 5.

2 Related Works and Contribution

Road traffic monitoring via the channel quality analysis in wireless network is a research topic of current interest. Several efforts have been accomplished to investigate the possibility of vehicles detection and classification with the use of channel state information (CSI) [5], the received signal strength indicator (RSSI) [6,7], a link quality indicator (LQI), and a packet loss rate [8].

A method which uses wireless transmission to detect road traffic congestion was proposed in [8]. This method requires a pair of wireless transmitter and receiver. The transmitter continuously sends packets. The receiver, which is placed on the opposite side of a road, evaluates RSSI, LQI, and packet loss metrics. It was shown that these metrics enable recognition between free-flow and congested traffic states with high accuracy. The method was implemented and tested with the use of ZigBee motes.

A similar ZigBee network was adapted in [7] for vehicles detection. Experimental results presented in that work confirm that a vehicle passing between the network nodes causes a drop of RSSI value. It was also observed that the gradient of the RSSI drop depends on the vehicle speed.

In [9] a method was introduced for vehicle detection and speed estimation, which is based on the RSSI analysis in a network composed of two WiFi access points and two WiFi-equipped laptops. The mean value and variance of RSSI measurements were used to discriminate between three states: empty road, stopped vehicle, and moving vehicle. The experimental results reported in [9] show that the variance of RSSI decreases with the increasing speed of a vehicle. This dependency was used for speed estimation.

Another WiFi-based traffic monitoring system was presented in [5]. This system utilizes a single access point and one laptop to provide functionalities of vehicle detection, classification, lane identification, and speed estimation. According to that approach, CSI patterns in WiFi network are captured and analyzed to perform the traffic monitoring tasks. The CSI characterizes signal strengths and phases of separate WiFi subcarriers. A machine learning technique was adopted to train vehicle classification models and categorize vehicles into two classes (passenger car and truck). It was also demonstrated that traffic lanes in a two-lane road have different distributions of CSI data. This fact was utilized to identify in which lane a vehicle is detected.

In [6] a radio-based approach for vehicle detection and classification was introduced, which combines ray tracing simulations, machine learning and RSSI measurements. The authors have suggested that different types of vehicles have specific RSSI fingerprints. This fact was used to perform a machine-based vehicle classification. The RSSI values were analyzed in a wireless network of three transmitting and three receiving units, which were positioned on opposite sides of a road. The six wireless units were mounted on delineator posts and equipped with directional antennas. It was demonstrated that such a system is able to detect vehicles and categorize them into two classes (passenger car and truck). The wireless networks have been also used for detection of parked vehicles. In order to detect the parked vehicles, the transmitting nodes are placed on a parking space and the receiving nodes are installed at a high location. When a vehicle is parked over the transmitting node, a decrease of the RSSI value is registered. Thus, the vehicles can be easily detected based on a simple RSSI analysis. Different systems of this type were implemented with the use of CC1101 wireless communication modules [10] and XBee motes [11].

Vehicle classification has been investigated using various machine learning methods: artificial neural networks [5], k-Nearest Neighbor (k-NN), support vector ma-chine (SVM) [6], decision tree [14], and logistic regression [16]. The algorithms have been usually trained using raw data [6] or using set of predefined features [14,16].

In this paper wireless vehicle detection and classification system is proposed, which performs the traffic monitoring tasks by analyzing strength of a radio signal received from Bluetooth Low Energy (BLE) beacons that are installed at different heights by the road. Contribution of this paper also includes a comparison of RSSI-based vehicles detection accuracy for different wireless communication technologies and development of methods that enable prolonging the lifetime of battery-powered wireless nodes in the traffic monitoring system.

3 Proposed Approach

In order to trace the changes of RSSI values that are caused by passing vehicles, the proposed traffic monitoring system uses a hybrid wireless network. The wireless nodes in this network are equipped with short range energy-efficient BLE communication and long range communication (GSM or WiFi). The BLE

communication is used to implement beacons that periodically broadcast data frames, as well as receivers that wait for incoming frames and evaluate RSSI upon frame arrival. The long range communication is used for transmitting the collected RSSI data to a server in a traffic monitoring center or a traffic light control unit. It should be noted that transmitting data in the same frequency as beacons could decrease the node ability to register incoming frames.

The BLE beacons use the iBeacon protocol [12] to broadcast the data frames. In particular, three fields in the iBeacon packet are used, which allows one to identify a beacon and its RSSI value: UUID (universally unique identifier), mayor and minor value.

Placement of the wireless nodes in a cross-section of the road is presented in Fig. 1. The wireless nodes (S) were installed at different heights for two reasons. Firstly, one of the research objectives was to analyze the dependency between the vertical placement of a BLE beacon-receiver pair and the accuracy of vehicles detection and classification. Secondly, arrangement of the nodes was motivated by the fact that height is a distinguishing feature of the vehicle classes that have to be recognized.

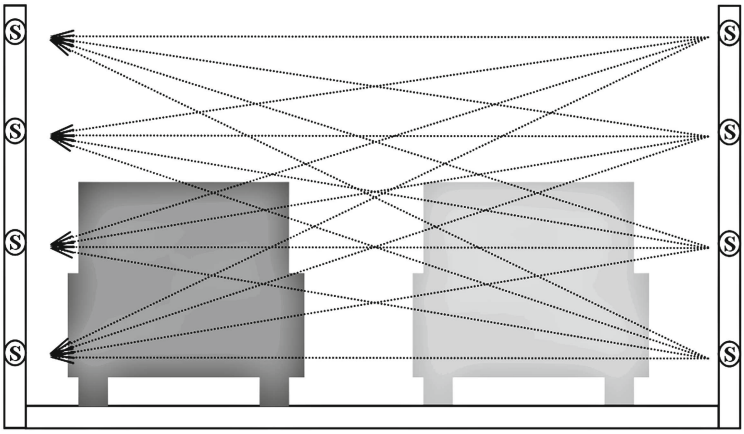


Fig. 1. The placement of wireless nodes (S)

It was assumed that the wireless nodes have recharging capabilities. Each node is able to work in two modes: as a BLE beacon (low energy mode) or as receiver, which collects RSSI data and reports it to the server via long range communication. Overview of the tasks performed by a wireless node in the proposed traffic monitoring system is presented in Fig. 2.

According to the scheme presented in Fig. 2 each node is initialised with the following information:

– id_gate - identifies the network and is defined in the UUID form for simplicity,

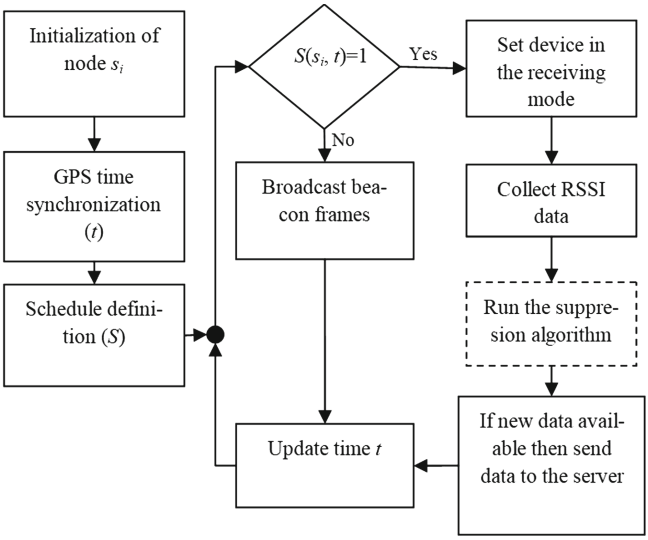


Fig. 2. Overview of tasks performed by a wireless node

- road_side $s_i \in \{0, 1\}$ - defines two groups of wireless nodes that are placed on opposite sides of the road (broadcasted as minor values),
- id_dev - node identifier equal to height (in centimetres) at which the node is installed (broadcasted as a mayor value),
- $op \in \{0, 1\}$ - determines if RSSI data should be sent to the server in real time (1) or in blocks (0),
- URL - address of the server in cloud.

It should be noted that initially the devices were identified by the MAC address. However, due to the anonymity policy, the MAC address can be modified randomly in modern devices to hide their identity. After initialisation, the node activates a GPS receiver to synchronise its real-time clock. The current time t in microseconds is represented as a long type value. Additionally, a request is sent to the server in order to obtain a schedule parameter t_{sh}.

It should be kept in mind that the wireless node consumes significantly more energy when operating in the receiver mode than in the beacon mode. To ensure balanced utilisation of the energy by the two groups of nodes (on both sides of the road) the beacon/receiver mode is switched according to schedule S, which is defined as follows:

$$S(s_i, t) = \begin{cases} 1 : \left\lfloor \frac{t}{t_{sh}} \right\rfloor \%2 = 1 \\ 0 : \quad else \end{cases} \tag{1}$$

where: t is the current time, $\lfloor x \rfloor$ denotes the floor operation of x, % is the modulo operation, and parameter t_{sh} (in microseconds) defines how often the beacon/receiver mode is switched. This schedule should be appropriately selected

to minimize data loss during the switching operation and to enable recharging the node. In practice it is easier to define the next switching time t_c and use it as a timer setting. The t_c value can be determined by using the formula:

$$t_c = t_{sh} - t\%t_{sh} \tag{2}$$

The iBeacon broadcast frequency can be adjusted from 1 Hz up to 10 Hz. However, if the device supports multiple beacons, this frequency can be multiplied. In order to determine the minimum broadcast frequency, which is necessary to detect a vehicle, the velocity of vehicle has to be taken under consideration. For a given maximum velocity ($vmax$ in km/h) and length of a vehicle (cl in meters) the minimum required frequency (in Hz) can be calculated as follows:

$$n_{min} = vmax/(3.6cl) \tag{3}$$

For $vmax = 100$ km/h and $cl = 2.8$ m the minimum frequency equals 9.25 Hz. In this study the experiments were conducted in a location where vehicles are moving at a speed not greater than 100 km/h. Thus, the broadcast frequency of 10 Hz was sufficient for the research purposes. The beacon broadcast power can also be modified. In this study the high transmission power was set in device (−56 dBm).

In case of the beacon mode the node is broadcasting beacons periodically and runs a timer, which triggers the switching to the receiver mode in accordance with the schedule (1). In the time between broadcasts the node is in the sleep state. To reduce the loss of beacon frames, the id_dev value was used as delay in microseconds when staring the beacon transmission routine. This allows beacons to broadcast frames at different times. In the receiver mode the antenna of BLE module is active all the time, which results in significant consumption of the energy resources.

In this study, initially performed research has shown that if the same communication interface is used to collect beacon frames and forward frames without a precise schedule, then significant frame loss is experienced. Thus, the second long range transmission module was provided. This module is an internet-of-things-type solution, which can communicate with a server via HTTP protocol. Two scenarios are considered regarding the data transmission to the server:

- real-time scenario, where data are used by a traffic light controller and data has to be forwarded to the controller with minimum delay,
- off-line scenario, where the traffic data can be send in blocks periodically and the transmission module can be activated only when a predetermined amount of data is collected.

The data exchange between the receiver node and the server is performed according to Algorithms 1 and 2 that are executed in parallel (as separate processes). The symbol d_{rssi} in Algorithm 1 denotes a threshold of RSSI change and each beacon frame b contains: network id (UUID), road side identifier (minor), and installation height (mayor).

In the proposed algorithm the data is transmitted to the server if the change of RSSI value is above the threshold d_{rssi}. This allows one to introduce a simple suppression of unnecessary data transmission.

Algorithm 1. Beacon frames processing at a receiver node

1. create empty array tabp
2. calculate tc using eq. (2), t_0 = current time,
3. while tc > current time- t_0 do
4. begin
5. if new frame b=($UUID_b$, $MINOR_b$, $MAYOR_b$) is received then
6. begin
7. evaluate $RSSI_b$
8. if $UUID_b$ = id_gate and $MINOR_b$ <> s_i then
9. if |tabp[$MAYOR_b$] - $RSSI_b$| > d_{rssi} or tabp[$MAYOR_b$] = null then
10. begin
11. if s_i = 0 then send frame s = ($UUID$, t, id_dev, $MAYOR_b$, $RSSI_b$) to buffer
12. else send frame s=($UUID$, t, $MAYOR_b$, id_dev, $RSSI_b$) to buffer
13. tabp[$MAYOR_b$] := $RSSI_b$
14. end
15. end
16. end

Algorithm 2. Data transmission from a receiver node to the server

1. while node is active do
2. case op of
3. 0: if buffer is full then send frames s from buffer to server
4. 1: if buffer is not empty then send frames s from buffer to server
5. end

The registered RSSI values are assigned to idealized paths of radio-signal transmission. These paths are described by ordered pairs of identifiers of the two nodes between which the signal is transmitted. First element of such a pair is an identifier of the node installed on side 0 ($s_i = 0$) and the second element is an identifier of the node installed on side 1 ($s_i = 1$). Lines 11 and 12 in Algorithm 1 present the location of the above-mentioned node identifiers (id_dev, $MAYOR_b$) inside the frame, which is sent to the server. The concept of signal paths is illustrated in Fig. 3. It should be kept in mind that the node identifier corresponds to height at which the device is installed. Thus, the RSSI values registered for transmissions in both directions between nodes placed at the same height are assigned to one path. However, in case when a beacon

and a receiver are at various heights, the transmissions in both directions are considered separately as their paths are different.

The transmission of collected RSSI data from a receiver node to the server is controlled by *op* parameter (Algorithm 2). In the case when $op = 1$ the data is sent with frequency of beacon broadcasts, otherwise the transmission is performed when buffer is full. Thus, for $op = 0$ the transmission frequency depends on the size of the buffer. The frame s is sent via http protocol in GET method. The server is performing data collection and classification based on provided data samples in accordance with Algorithm 2, where sampling rate sr is defined in microseconds.

Algorithm 3. Data processing at the server

1. create table TAB with columns $UUID$, t, p_start, p_end, $RSSI$
2. at each time step (t) do
3. if new frame s=($UUID_s$, t_s, p_start_s, p_end_s, $RSSI_s$) is received then
4. begin
5. insert into TAB values ($UUID_s$, t_s, p_start_s, p_end_s, $RSSI_s$)
6. V := select * from TAB where ($UUID$ = $UUID_s$) and (t between $t_s - sr$ and t_s)
7. run classification for V
8. end

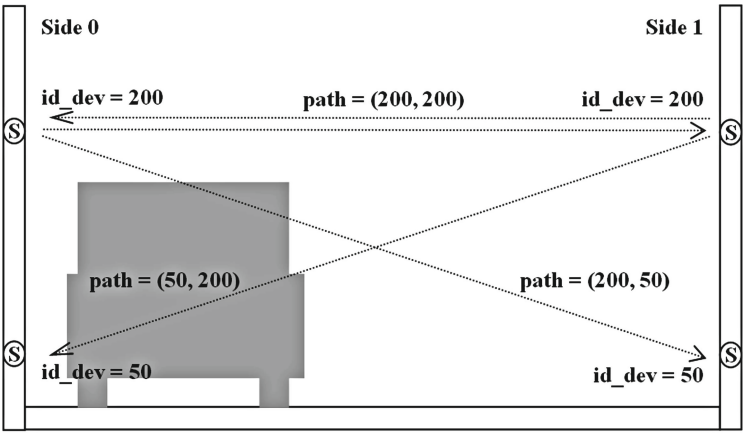


Fig. 3. Idealized paths of radio signal transmissions

The objective of the classification procedure (line 7 in Algorithm 3) is to recognize if a car was present in the region of interest during time interval $[t_s - sr; t_s]$. Once vehicle is detected, the classification procedure has to determine its class (personal car, semi-truck, or truck). At the beginning of this procedure,

the input data set V is preprocessed to complete suppressed RSSI values. In this study, the classification procedure was implemented in three versions, with the use of three different machine learning algorithms (random forest, probabilistic neural network, and k-nearest neighbors' algorithm) [13,14]. A separate data set, which includes vehicle classes deter-mined by a human observer (personal car, semi-truck and truck), was used to train the classifiers.

4 Experiments

Initial research was conducted to analyze the usefulness of different types of wireless devices in vehicle detection and classification. The three most popular short-range wireless communication technologies were taken into consideration: ZigBee (Xbee S2C DigiMesh 2.4), BLE (Qualcomm MSM8937 Snapdragon 430 with BLE 4.1 support), and IQRF (IQRF TR-52D). During experiments the two communicating devices were placed 50 cm above the road surface. The vehicles were classified every second. All the devices were transmitting frames with the same frequency (10 Hz). Specific parameters of the wireless devices are presented in Table 1. Results describing detection accuracy for 60 vehicles are presented in Table 2. These results were obtained by using three different classification algorithms: k-nearest neighbors algorithm (KNN), random forest [17] (RF), and multi-layer probabilistic neural networks (PNN).

Table 1. Parameters of wireless devices used for RSSI measurements

Parameter	IQRF	Zigbee	BLE
Transmission power	Lowest	Lowest	Lowest
Broadcast frequency	ISM 868-870 MHz	ISM 2.4 GHz	ISM 2.4 GHz
Antenna RSSI value collection	PCB using AT commands	PCB reading value using PWN pin	PCB reading value via internal structure

The detection accuracy was calculated as overall accuracy using the following formula:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} 100\% \tag{4}$$

where: TP - number of true positives, TN - number of true negatives, FP - number of false positives, and FN - number of false negatives. In this research, the samples with vehicle passage were compared with samples of empty road. Based on the preliminary results (Table 2), BLE devices were selected as giving high detection accuracy for all considered classification algorithms.

For further experiments, a prototype of the proposed traffic monitoring system was created. It consists of the beacon and receiver modules, which were

Table 2. Accuracy of vehicles detection with the use of various wireless technologies

Type of device	Classifier	Detection accuracy (%)
IQRF	KNN	60.00
	RF	69.20
	PNN	63.60
ZigBee	KNN	74.50
	RF	75.70
	PNN	79.80
BLE	KNN	77.50
	RF	78.70
	PNN	78.80

implemented with the use of Redmi 3S mobile device, which is equipped with Qualcomm MSM8937 Snapdragon 430, 4100 mAh battery, and enables communication via BLE, Wi-Fi, and GSM modem.

Examples of registered RSSI profiles are illustrated in Fig. 4. The vertical lines in Fig. 4 show the time periods when vehicles are present between the wireless nodes. These results show that RSSI values collected by devices installed close to road surface are not useful for vehicle detection. In the case when the wireless nodes are placed on the pavement (Fig. 4a) the vehicles do not cause visible changes of RSSI because of signal tunneling under vehicle chassis and relatively low impact of wheels. The clear RSSI changes can be observed when the wireless nodes are installed 50 cm above the road (Fig. 4b).

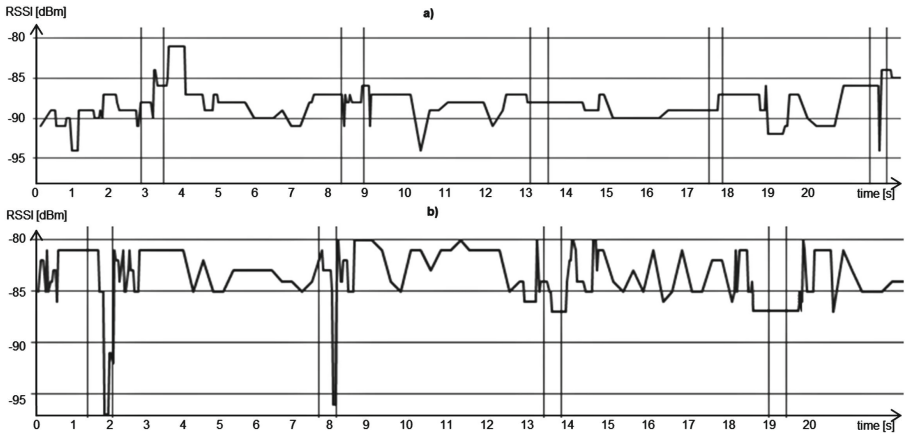


Fig. 4. Changes of RSSI value for passing vehicles for wireless devices installed at: (a) 0 cm, (b) 50 cm

The possibility of vehicle detection and classification was verified for the BLE nodes installed in various height configurations. The registered changes of RSSI values for various configurations are illustrated by examples in Fig. 5. The pairs of values presented above the charts in Fig. 5 denote installation heights of beacon (first value) and receiver node (second value).

The results presented in Fig. 5 confirm that the low localization of a beacon and a receiver (10 cm–10 cm) is inappropriate for vehicle detection. In this configuration, single RSSI variations are caused by wheels of the moving vehicles. However, these variations are insufficient for reliable detection and classification. What is worth noting, a passing vehicle (marked as two vertical lines in Fig. 4) can cause an increase of the signal strength, which makes the reasoning even more difficult.

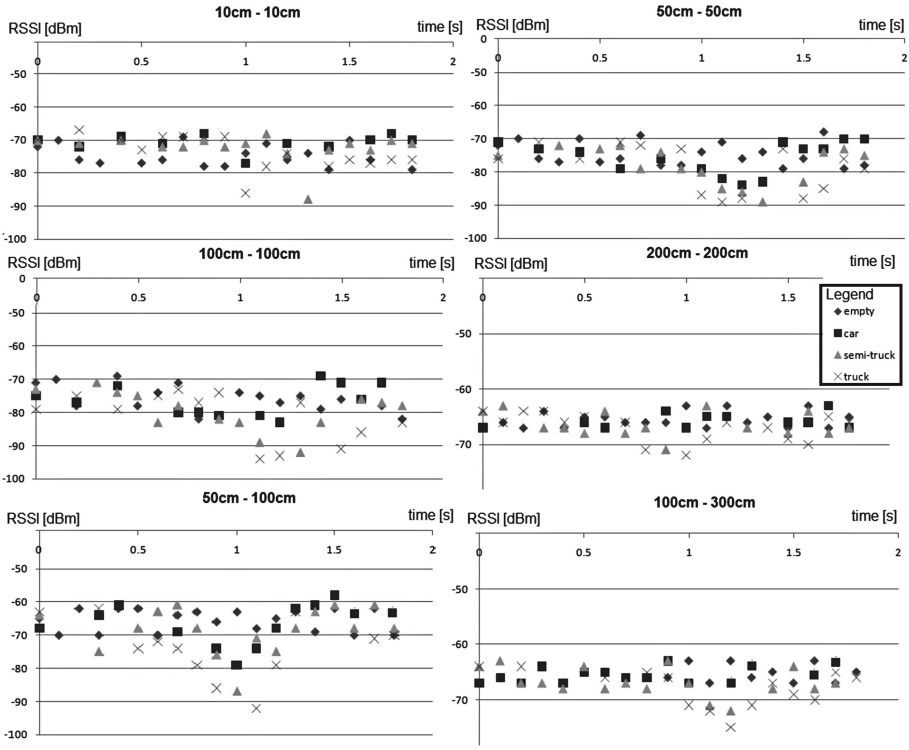


Fig. 5. RSSI changes caused by vehicle pass for various receiver-beacon height configurations

The strongest changes of RSSI values were obtained for configuration 50 cm–50 cm, i.e., for both nodes installed at the height of 50 cm. It was also observed that the noise (reflection of a signal from the ground and a vehicles chassis) in RSSI measurement is reduced when increasing the distance from the ground.

However, the RSSI difference during vehicle pass also decreases significantly with increasing height. The placement of a beacon and a receiver at height above 100 cm is suitable for detecting trucks and semi-trucks. It was also noticed that improved results can be obtained when a transmitter and a receiver are installed at different heights. For instance, trucks can be easily recognized based on the RSSI changes when one node is relatively close to the ground (50 cm) and the other one is above 100 cm.

During experiments, it was observed that not all beacon frames are received. Thus, additional laboratory measurements were conducted to check the ability of the receiver to collect beacon frames. The data was registered for 15 min. Fig. 6 presents box plot (minimum, maximum, first and third quartile) of numbers of beacon frames that were received within one second intervals for different beacon-receiver distances and various frame broadcast frequencies.

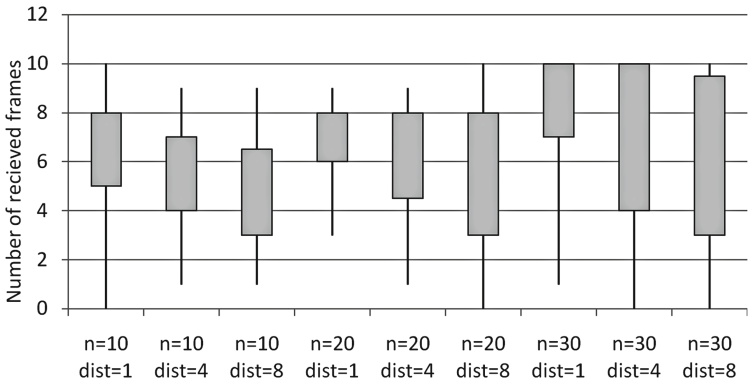


Fig. 6. The box plot for frames received within one second interval for different beacon-receiver distances (dist in meters) and various broadcast frequencies (n in Hz)

The results in Fig. 6 firmly show that the BLE module used in the experiments cannot collect more than 10 frames per second. The application of multiple receiver nodes in the proposed network allows one to overcome these limitations. Accuracy of vehicle detection and classification obtained for a single pair of the wireless nodes and for various combinations of their installation heights (scenarios) is presented in Table 3. The experiment was performed for 60 vehicles. Three vehicle classes were considered: personal car, semi-truck and truck. The vehicle detection and classification tasks were accomplished using various machine learning methods (RF, PNN, and k-NN). The results in Table 3 show that for one beacon-receiver pair the maximum achieved detection accuracy equals 81%, and the maximum classification accuracy was 74%. It can be also observed that the detection and classification accuracy is decreased if a beacon and a receiver are installed at the height of 300 cm. Therefore, the heights between 50 cm and 200 cm were considered in further experiments. In order to improve the accuracy,

a wireless network was used with six nodes. Three nodes were placed at height of 50 cm, 100 cm, and 200 cm on both sides of the road. The results for this network are presented in Table 4.

Table 3. Accuracy of vehicle detection and classification for various scenarios with two nodes (x- receiver height [cm], y- transmiter height [cm])

Scen. x_y	Classifier	Detection	Classification	Scen. x_y	Classifier	Detection	Classification
50_50	RF	81%	74%	100_200	RF	65%	58%
	PNN	79%	71%		PNN	69%	60%
	kNN	80%	72%		kNN	65%	61%
50_100	RF	78%	70%	100_300	RF	78%	70%
	PNN	73%	65%		PNN	76%	68%
	kNN	75%	66%		kNN	69%	63%
50_200	RF	71%	62%	200_200	RF	59%	52%
	PNN	73%	67%		PNN	64%	56%
	kNN	65%	61%		kNN	57%	50%
50_300	RF	57%	51%	200_300	RF	58%	54%
	PNN	57%	51%		PNN	61%	57%
	kNN	53%	48%		kNN	55%	53%
100_100	RF	80%	73%	300_300	RF	52%	55%
	PNN	78%	68%		PNN	57%	57%
	kNN	77%	71%		kNN	54%	54%

Table 4 presents the results that were obtained by using various classifiers, including those, which were applied for RSSI-based vehicle detection in previous works [6,14,16]. The classification was performed using KNIME environment with Weka tools [15]. In case of RF algorithm 100 models were used with information gain ratio as split criteria. The PNN classifier is based on the dynamic decay adjustment method (*theta* parameter was in range between 0.2 and 0.4). In case of k-NN algorithm, the k parameter was set to 3. The LS-SVM algorithm corresponds to linear kernel implementation of nu-SVC (with $nu = 0.5$). The DT classifier uses C4.5 algorithm (gain ratio was used as quality measure without pruning). Cohen's kappa was taken into account as an additional measure of performance, as it is robust to class imbalances [14]. It was calculated based on approach proposed in [18].

It should be noted that utilization of the RSSI data collected by three receiver nodes allows algorithm to improve accuracy for detection and the classification. The vehicle classification and detection tasks were executed every second ($sr =$ 1s) and the threshold of RSSI change for data suppression (d_{rssi}) was set to 4 dBm based on preliminary experiments.

The experiments were also conducted to evaluate energy consumption and lifetime of the wireless node. During these experiments, the node was used with and without solar panel (Fig. 7). The energy consumption was determined during 6-hour test on the road in daylight conditions: mostly a sunny day with aver-

Table 4. Accuracy of vehicle detection and classification for various scenarios with two nodes

Classifier	Detection		Classification	
	Accuracy	Cohen's kappa	Accuracy	Cohen's kappa
RF	98%	97%	97%	96%
PNN	95%	90%	91%	88%
kNN	95%	93%	94%	90%
LS-SVM	92%	85%	95%	92%
Logistic Regression	91%	83%	87%	81%
DT	94%	92%	88%	85%

age temperature equal to 4 °C. The energy consumption (at 4.4 V) and lifetime evaluated for different configurations of the node are presented in Table 5.

Fig. 7. Wireless node with solar panel and the management application

Table 5. Energy consumption and lifetime of the wireless node without solar panel

Schedule	Node configuration	Lifetime (h)	Energy consumption (mA)
No	Beacon	1250	3.28
No	Receiver real-time (op = 1)	35.7	114.8
No	Receiver off-line (op = 0)	62.5	65.6
Yes	Beacon/Receiver real-time (op = 1)	66.5	61.6
Yes	Beacon/Receiver off-line (op = 0)	116.8	35.1

Subsequently, the experiment was performed using the receiver node with a solar panel and the beacon node. Application of the rechargeable device presented in Fig. 7 (60 mA charge current) allows the receiver node to work for 14 days and retain no less than 50% of battery energy in off-line mode ($op = 0$). In the case of the real-time mode ($op = 1$) the panel was insufficient to

recharge the receiver node. Thus, in future implementation a solar panel with better energy capabilities will be used. It is also worth mentioning that additional energy consumption was caused by the Android operating system.

5 Conclusion

The proposed wireless vehicle detection and classification system allows one to perform the traffic monitoring tasks with the accuracy above 97% using the random forest classifier. It is detecting three classes of vehicles by analyzing strength of radio signal received from BLE beacons that are installed at different heights by the road. The extended analysis showed relation between the height at which the devices are installed and the ability to detect particular vehicle class.

The paper compares RSSI-based vehicles detection accuracy for different wireless communication technologies. The results show that Zigbee and BLE standards are effective in the RSSI-based traffic monitoring applications. Additionally, the proposed method uses schedule and role changes to extend the lifetime of battery-powered wireless nodes in the traffic monitoring system. The results show that this solution extends lifetime of the network twice and enables better utilization of the recharging mechanism. The implemented photo-voltaic recharging device allowed the lifetime of the network to be extended even further (over two weeks). The proposed BLE-based solution:

- collects data in real time for traffic control and for statistical purposes,
- uses schedules, that enables uniform consumption of energy resources in nodes,
- provides high vehicle classification accuracy for two-lane road,
- involves a recharge mechanism to extend the network lifespan.

The presented wireless network could be enhanced in the future by introducing a more sophisticated beacon broadcasting mechanism to decrease the number of frame collisions. The solution can be extended to several measuring points along a road to obtain information concerning vehicle velocity and direction. Another research topic could focus on data aggregation methods to simplify the classification process. Finally, additional sensor data can be used to activate the receiver node when it is necessary and reduce the energy consumption.

Acknowledgement. The research was supported by The National Centre for Research and Development (NCBR) grant number LIDER/18/0064/L-7/15/NCBR/2016.

References

1. Bernas, M., Płaczek, B., Porwik, P., Pamuła, T.: Segmentation of vehicle detector data for improved K-nearest neighbours-based traffic flow prediction. IET Intell. Transp. Syst. IET Intell. Transp. Syst. **9**(3), 264–274 (2015)
2. Piorkowski, A.: Construction of a Dynamic Arrival Time Coverage Map for Emergency Medical Services. Open Geosciences (in Print)
3. Płaczek, B.: Selective data collection in vehicular networks for traffic control applications. Transp. Res. Part C Emerg. Technol. **23**, 14–28 (2012)
4. Bernas, M., Płaczek, B.: Zone-based VANET transmission model for traffic signal control. In: Gaj, P., Kwiecień, A., Sawicki, M. (eds.) CN 2017. CCIS, vol. 718, pp. 444–457. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59767-6_35
5. Won, M., Zhang, S., Son, S.H.: WiTraffic: low-cost and non-intrusive traffic monitoring system using WiFi. In: 26th International Conference on Computer Communication and Networks (ICCCN), pp. 1–9 (2017)
6. Haferkamp, M., Al-Askary, M., Dorn, D., Sliwa, B., Habel, L., Schreckenberg, M., Wietfeld, C.: Radio-based traffic flow detection and vehicle classification for future smart cities. In: IEEE 85th Vehicular Technology Conference, pp. 1–5 (2017)
7. Horvat, G., Sostaric, D., Zagar, D.: Using radio irregularity for vehicle detection in adaptive roadway lighting. In: 35th International Convention, MIPRO 2012, pp. 748–753 (2012)
8. Roy, S., Sen, R., Kulkarni, S., Kulkarni, P., Raman, B., Singh, L.K.: Wireless across road: RF based road traffic congestion detection. In: 3rd International Conference on Communication Systems and Networks (COMSNETS), pp. 1–6 (2011)
9. Kassem, N., Kosba, A.E., Youssef, M.: RF-based vehicle detection and speed estimation. In: IEEE 75th Vehicular Technology Conference, pp. 1–5 (2012)
10. Li, X., Wu, J.: A new method and verification of vehicles detection based on RSSI variation. In: 10th International Conference on Sensing Technology (ICST), pp. 1–6 (2016)
11. Mestre, P., Guedes, R., Couto, P., Matias, J., Fernandes, J.C., Serodio, C: Vehicle detection for outdoor car parks using IEEE 802.15.4. Lecture Notes in Engineering and Computer Science. Newswood Limited-IAENG (2013)
12. Apple Inc.: Getting Started with iBeacon. Technical Report 1.0, June 2014
13. Berthold, M.R. et al.: KNIME: the konstanz information miner. In: Preisach, C., Burkhardt, H., Schmidt-Thieme, L., Decker, R. (eds.) Data Analysis, Machine Learning and Applications. Studies in Classification, Data Analysis, and Knowledge Organization. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78246-9_38
14. Martsenyuk, V. et al.: On multivariate method of qualitative analysis of hodgkin-huxley model with decision tree induction. In: 16th International Conference on Control, Automation and Systems (ICCAS), pp. 489–494 (2016)
15. Smeeton, N.C.: Early history of the kappa statistic. Biometrics **41**, 795 (1985)
16. Kleyko, D., Hostettler, R., Birk, W., Osipov, E.: Comparison of machine learning techniques for vehicle classification using road side sensors. In: IEEE 18th International Conference Intelligent Transportation Systems, pp. 572–577 (2015)
17. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
18. Bakeman, R., Gottman, J.M.: Observing Interaction: An Introduction to Sequential Analysis, 2nd edn. Cambridge University Press, Cambridge (1997)

Author Index