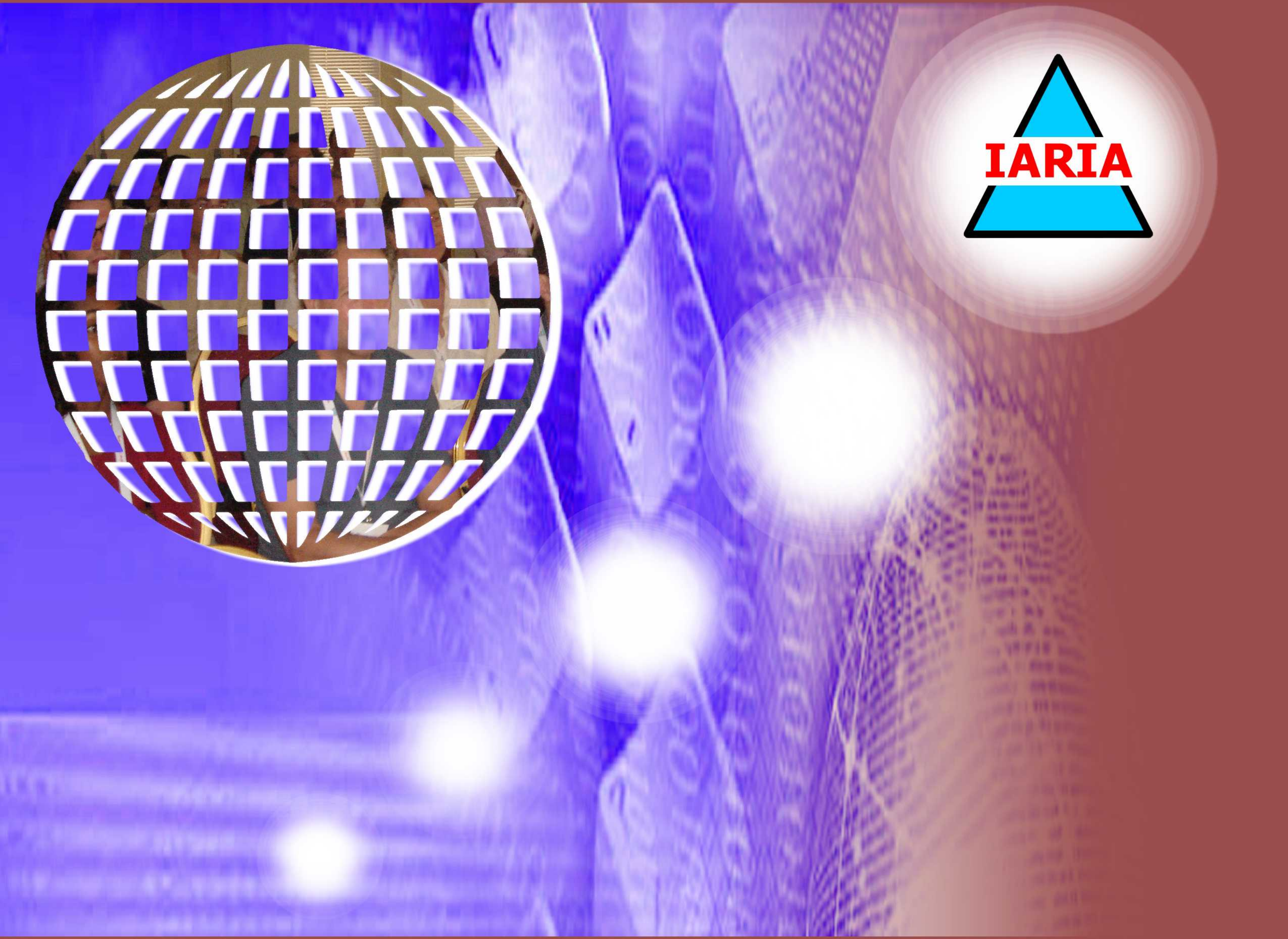
International Journal on Advances in Networks and Services



2019 vol. 12 nr. 1&2

The *International Journal on Advances in Networks and Services* is published by IARIA.
ISSN: 1942-2644
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Responsibility for the contents rests upon the authors and not upon IARIA, nor on IARIA volunteers, staff, or contractors.

IARIA is the owner of the publication and of editorial aspects. IARIA reserves the right to update the content for quality improvements.

Abstracting is permitted with credit to the source. Libraries are permitted to photocopy or print, providing the reference is mentioned and that the resulting material is made available at no cost.

Reference should mention:

International Journal on Advances in Networks and Services, issn 1942-2644
vol. 12, no. 1 & 2, year 2019, http://www.iariajournals.org/networks_and_services/

The copyright for each included paper belongs to the authors. Republishing of same material, by authors or persons or organizations, is not allowed. Reprint rights can be granted by IARIA or by the authors, and must include proper reference.

Reference to an article in the journal is as follows:

<Author list>, "<Article title>"
International Journal on Advances in Networks and Services, issn 1942-2644
vol. 12, no. 1 & 2, year 2019, <start page>:<end page> , http://www.iariajournals.org/networks_and_services/

IARIA journals are made available for free, proving the appropriate references are made when their content is used.

Sponsored by IARIA
www.iaria.org

Copyright © 2019 IARIA

International Journal on Advances in Networks and Services
Volume 12, Number 1 & 2, 2019

Editor-in-Chief

Tibor Gyires, Illinois State University, USA

Editorial Advisory Board

Mario Freire, University of Beira Interior, Portugal
Carlos Becker Westphall, Federal University of Santa Catarina, Brazil
Rainer Falk, Siemens AG - Corporate Technology, Germany
Cristian Anghel, University Politehnica of Bucharest, Romania
Rui L. Aguiar, Universidade de Aveiro, Portugal
Jemal Abawajy, Deakin University, Australia
Zoubir Mammeri, IRIT - Paul Sabatier University - Toulouse, France

Editorial Board

Ryma Abassi, Higher Institute of Communication Studies of Tunis (Iset'Com) / Digital Security Unit, Tunisia
Majid Bayani Abbasy, Universidad Nacional de Costa Rica, Costa Rica
Jemal Abawajy, Deakin University, Australia
Javier M. Aguiar Pérez, Universidad de Valladolid, Spain
Rui L. Aguiar, Universidade de Aveiro, Portugal
Ali H. Al-Bayati, De Montfort Uni. (DMU), UK
Giuseppe Amato, Consiglio Nazionale delle Ricerche, Istituto di Scienza e Tecnologie dell'Informazione (CNR-ISTI), Italy
Mario Anzures-García, Benemérita Universidad Autónoma de Puebla, México
Pedro Andrés Aranda Gutiérrez, Telefónica I+D - Madrid, Spain
Cristian Anghel, University Politehnica of Bucharest, Romania
Miguel Ardid, Universitat Politècnica de València, Spain
Valentina Baljak, National Institute of Informatics & University of Tokyo, Japan
Alvaro Barradas, University of Algarve, Portugal
Mostafa Bassiouni, University of Central Florida, USA
Michael Bauer, The University of Western Ontario, Canada
Carlos Becker Westphall, Federal University of Santa Catarina, Brazil
Zdenek Becvar, Czech Technical University in Prague, Czech Republic
Francisco J. Bellido Outeiriño, University of Cordoba, Spain
Djamel Benferhat, University Of South Brittany, France
Jalel Ben-Othman, Université de Paris 13, France
Mathilde Benveniste, En-aerion, USA
Luis Bernardo, Universidade Nova of Lisboa, Portugal
Alex Bikfalvi, Universidad Carlos III de Madrid, Spain
Thomas Michael Bohnert, Zurich University of Applied Sciences, Switzerland
Eugen Borgoci, University "Politehnica"of Bucharest (UPB), Romania

Fernando Boronat Seguí, Universidad Politecnica de Valencia, Spain
Christos Bouras, University of Patras, Greece
Mahmoud Brahimi, University of Msila, Algeria
Marco Bruti, Telecom Italia Sparkle S.p.A., Italy
Dumitru Burdescu, University of Craiova, Romania
Diletta Romana Cacciagrano, University of Camerino, Italy
Maria-Dolores Cano, Universidad Politécnica de Cartagena, Spain
Juan-Vicente Capella-Hernández, Universitat Politècnica de València, Spain
Eduardo Cerqueira, Federal University of Para, Brazil
Bruno Chatras, Orange Labs, France
Marc Cheboldaeff, Deloitte Consulting GmbH, Germany
Kong Cheng, Vencore Labs, USA
Dickson Chiu, Dickson Computer Systems, Hong Kong
Andrzej Chydzinski, Silesian University of Technology, Poland
Hugo Coll Ferri, Polytechnic University of Valencia, Spain
Noelia Correia, University of the Algarve, Portugal
Noël Crespi, Institut Telecom, Telecom SudParis, France
Paulo da Fonseca Pinto, Universidade Nova de Lisboa, Portugal
Orhan Dagdeviren, International Computer Institute/Ege University, Turkey
Philip Davies, Bournemouth and Poole College / Bournemouth University, UK
Carlton Davis, École Polytechnique de Montréal, Canada
Claudio de Castro Monteiro, Federal Institute of Education, Science and Technology of Tocantins, Brazil
João Henrique de Souza Pereira, University of São Paulo, Brazil
Javier Del Ser, Tecnalia Research & Innovation, Spain
Behnam Dezfouli, Universiti Teknologi Malaysia (UTM), Malaysia
Daniela Dragomirescu, LAAS-CNRS, University of Toulouse, France
Jean-Michel Dricot, Université Libre de Bruxelles, Belgium
Wan Du, Nanyang Technological University (NTU), Singapore
Matthias Ehmann, Universität Bayreuth, Germany
Wael M El-Medany, University Of Bahrain, Bahrain
Imad H. Elhajj, American University of Beirut, Lebanon
Gledson Elias, Federal University of Paraíba, Brazil
Joshua Ellul, University of Malta, Malta
Rainer Falk, Siemens AG - Corporate Technology, Germany
Károly Farkas, Budapest University of Technology and Economics, Hungary
Huei-Wen Ferng, National Taiwan University of Science and Technology - Taipei, Taiwan
Gianluigi Ferrari, University of Parma, Italy
Mário F. S. Ferreira, University of Aveiro, Portugal
Bruno Filipe Marques, Polytechnic Institute of Viseu, Portugal
Ulrich Flegel, HFT Stuttgart, Germany
Juan J. Flores, Universidad Michoacana, Mexico
Ingo Friese, Deutsche Telekom AG - Berlin, Germany
Sebastian Fudickar, University of Potsdam, Germany
Stefania Galizia, Innova S.p.A., Italy
Ivan Ganchev, University of Limerick, Ireland / University of Plovdiv “Paisii Hilendarski”, Bulgaria
Miguel Garcia, Universitat Politecnica de Valencia, Spain

Emiliano Garcia-Palacios, Queens University Belfast, UK
Marc Gilg, University of Haute-Alsace, France
Debasis Giri, Haldia Institute of Technology, India
Markus Goldstein, Kyushu University, Japan
Luis Gomes, Universidade Nova Lisboa, Portugal
Anahita Gouya, Solution Architect, France
Mohamed Graiet, Institut Supérieur d'Informatique et de Mathématique de Monastir, Tunisie
Christos Grecos, University of West of Scotland, UK
Vic Grout, Glyndwr University, UK
Yi Gu, Middle Tennessee State University, USA
Angela Guercio, Kent State University, USA
Xiang Gui, Massey University, New Zealand
Mina S. Guirguis, Texas State University - San Marcos, USA
Tibor Gyires, School of Information Technology, Illinois State University, USA
Keijo Haataja, University of Eastern Finland, Finland
Gerhard Hancke, Royal Holloway / University of London, UK
R. Hariprakash, Arulmigu Meenakshi Amman College of Engineering, Chennai, India
Go Hasegawa, Osaka University, Japan
Eva Hladká, CESNET & Masaryk University, Czech Republic
Hans-Joachim Hof, Munich University of Applied Sciences, Germany
Razib Iqbal, Amdocs, Canada
Abhaya Induruwa, Canterbury Christ Church University, UK
Muhammad Ismail, University of Waterloo, Canada
Vasanth Iyer, Florida International University, Miami, USA
Imad Jawhar, United Arab Emirates University, UAE
Aravind Kailas, University of North Carolina at Charlotte, USA
Mohamed Abd rabou Ahmed Kalil, Ilmenau University of Technology, Germany
Kyoung-Don Kang, State University of New York at Binghamton, USA
Sarfraz Khokhar, Cisco Systems Inc., USA
Vitaly Klyuev, University of Aizu, Japan
Jarkko Kneckt, Nokia Research Center, Finland
Dan Komosny, Brno University of Technology, Czech Republic
Ilker Korkmaz, Izmir University of Economics, Turkey
Tomas Koutny, University of West Bohemia, Czech Republic
Evangelos Kranakis, Carleton University - Ottawa, Canada
Lars Krueger, T-Systems International GmbH, Germany
Kae Hsiang Kwong, MIMOS Berhad, Malaysia
KP Lam, University of Keele, UK
Birger Lantow, University of Rostock, Germany
Hadi Larijani, Glasgow Caledonian Univ., UK
Annett Laube-Rosenpflanzer, Bern University of Applied Sciences, Switzerland
Gyu Myoung Lee, Institut Telecom, Telecom SudParis, France
Shiguo Lian, Orange Labs Beijing, China
Chiu-Kuo Liang, Chung Hua University, Hsinchu, Taiwan
Wei-Ming Lin, University of Texas at San Antonio, USA
David Lizcano, Universidad a Distancia de Madrid, Spain

Chengnian Long, Shanghai Jiao Tong University, China
Jonathan Loo, Middlesex University, UK
Pascal Lorenz, University of Haute Alsace, France
Albert A. Lysko, Council for Scientific and Industrial Research (CSIR), South Africa
Pavel Mach, Czech Technical University in Prague, Czech Republic
Elsa María Macías López, University of Las Palmas de Gran Canaria, Spain
Damien Magoni, University of Bordeaux, France
Ahmed Mahdy, Texas A&M University-Corpus Christi, USA
Zoubir Mammeri, IRIT - Paul Sabatier University - Toulouse, France
Gianfranco Manes, University of Florence, Italy
Sathiamoorthy Manoharan, University of Auckland, New Zealand
Moshe Timothy Masonta, Council for Scientific and Industrial Research (CSIR), Pretoria, South Africa
Hamid Menouar, QU Wireless Innovations Center - Doha, Qatar
Guowang Miao, KTH, The Royal Institute of Technology, Sweden
Mohssen Mohammed, University of Cape Town, South Africa
Miklos Molnar, University Montpellier 2, France
Lorenzo Mossucca, Istituto Superiore Mario Boella, Italy
Jogesh K. Muppala, The Hong Kong University of Science and Technology, Hong Kong
Katsuhiro Naito, Mie University, Japan
Deok Hee Nam, Wilberforce University, USA
Sarmistha Neogy, Jadavpur University- Kolkata, India
Rui Neto Marinheiro, Instituto Universitário de Lisboa (ISCTE-IUL), Instituto de Telecomunicações, Portugal
David Newell, Bournemouth University - Bournemouth, UK
Ngoc Tu Nguyen, Missouri University of Science and Technology - Rolla, USA
Armando Nolasco Pinto, Universidade de Aveiro / Instituto de Telecomunicações, Portugal
Jason R.C. Nurse, University of Oxford, UK
Kazuya Odagiri, Yamaguchi University, Japan
Máirtín O'Droma, University of Limerick, Ireland
Jose Oscar Fajardo, University of the Basque Country, Spain
Constantin Paleologu, University Politehnica of Bucharest, Romania
Eleni Patouni, National & Kapodistrian University of Athens, Greece
Harry Perros, NC State University, USA
Miodrag Potkonjak, University of California - Los Angeles, USA
Yusnita Rahayu, Universiti Malaysia Pahang (UMP), Malaysia
Yenumula B. Reddy, Grambling State University, USA
Oliviero Riganelli, University of Milano Bicocca, Italy
Antonio Ruiz Martinez, University of Murcia, Spain
George S. Oreku, TIRDO / North West University, Tanzania/ South Africa
Sattar B. Sadkhan, Chairman of IEEE IRAQ Section, Iraq
Husnain Saeed, National University of Sciences & Technology (NUST), Pakistan
Addisson Salazar, Universidad Politecnica de Valencia, Spain
Sébastien Salva, University of Auvergne, France
Ioakeim Samaras, Aristotle University of Thessaloniki, Greece
Luz A. Sánchez-Gálvez, Benemérita Universidad Autónoma de Puebla, México
Teerapat Sanguankotchakorn, Asian Institute of Technology, Thailand
José Santa, University Centre of Defence at the Spanish Air Force Academy, Spain

Rajarshi Sanyal, Belgacom International Carrier Services, Belgium
Mohamad Sayed Hassan, Orange Labs, France
Thomas C. Schmidt, HAW Hamburg, Germany
Véronique Sebastien, University of Reunion Island, France
Jean-Pierre Seifert, Technische Universität Berlin & Telekom Innovation Laboratories, Germany
Dimitrios Serpanos, Univ. of Patras and ISI/RC ATHENA, Greece
Roman Y. Shtykh, Rakuten, Inc., Japan
Salman Ijaz Institute of Systems and Robotics, University of Algarve, Portugal
Adão Silva, University of Aveiro / Institute of Telecommunications, Portugal
Florian Skopik, AIT Austrian Institute of Technology, Austria
Karel Slavicek, Masaryk University, Czech Republic
Vahid Solouk, Urmia University of Technology, Iran
Peter Soreanu, ORT Braude College, Israel
Pedro Sousa, University of Minho, Portugal
Cristian Stanciu, University Politehnica of Bucharest, Romania
Vladimir Stantchev, SRH University Berlin, Germany
Radu Stoleru, Texas A&M University - College Station, USA
Lars Strand, Nofas, Norway
Stefan Strauβ, Austrian Academy of Sciences, Austria
Álvaro Suárez Sarmiento, University of Las Palmas de Gran Canaria, Spain
Masashi Sugano, School of Knowledge and Information Systems, Osaka Prefecture University, Japan
Young-Joo Suh, POSTECH (Pohang University of Science and Technology), Korea
Junzhao Sun, University of Oulu, Finland
David R. Surma, Indiana University South Bend, USA
Yongning Tang, School of Information Technology, Illinois State University, USA
Yoshiaki Taniguchi, Kindai University, Japan
Anel Tanovic, BH Telecom d.d. Sarajevo, Bosnia and Herzegovina
Rui Teng, Advanced Telecommunications Research Institute International, Japan
Olivier Terzo, Istituto Superiore Mario Boella - Torino, Italy
Tzu-Chieh Tsai, National Chengchi University, Taiwan
Samyr Vale, Federal University of Maranhão - UFMA, Brazil
Dario Vieira, EFREI, France
Lukas Vojtech, Czech Technical University in Prague, Czech Republic
Michael von Riegen, University of Hamburg, Germany
You-Chiun Wang, National Sun Yat-Sen University, Taiwan
Gary R. Weckman, Ohio University, USA
Chih-Yu Wen, National Chung Hsing University, Taichung, Taiwan
Michelle Wetterwald, HeNetBot, France
Feng Xia, Dalian University of Technology, China
Kaiping Xue, USTC - Hefei, China
Mark Yampolskiy, Vanderbilt University, USA
Dongfang Yang, National Research Council, Canada
Qimin Yang, Harvey Mudd College, USA
Beytullah Yildiz, TOBB Economics and Technology University, Turkey
Anastasiya Yurchyshyna, University of Geneva, Switzerland
Sergey Y. Yurish, IFSA, Spain

Jelena Zdravkovic, Stockholm University, Sweden
Yuanyuan Zeng, Wuhan University, China
Weiliang Zhao, Macquarie University, Australia
Wenbing Zhao, Cleveland State University, USA
Zibin Zheng, The Chinese University of Hong Kong, China
Yongxin Zhu, Shanghai Jiao Tong University, China
Zuqing Zhu, University of Science and Technology of China, China
Martin Zimmermann, University of Applied Sciences Offenburg, Germany

International Journal on Advances in Networks and Services
Volume 12, Numbers 1 & 2, 2019

CONTENTS

pages: 1 - 9
A Study on Round-trip Time Estimation from Unidirectional Packet Traces Using Different TCP Congestion Control Algorithms
Toshihiko Kato, University of Electro-Communications, Japan
Xiaofan Yan, University of Electro-Communications, Japan
Ryo Yamamoto, University of Electro-Communications, Japan
Satoshi Ohzahata, University of Electro-Communications, Japan

pages: 10 - 20
Experimental Modeling and Measurements of Networked Wireless Sensors for Power Consumption
Jin-Shyan Lee, National Taipei University of Technology, Taiwan
Jorge Portilla, Universidad Politecnica de Madrid, Spain
Gabriel Mujica, Universidad Politecnica de Madrid, Spain
Yuan-Heng Sun, Industrial Technology Research Institute, Taiwan

pages: 21 - 29
Optimal Control of Multicasting Multimedia Streams in Cloud Data Centers Networks
Nader Mir, San Jose State University, California, U.S.A
Abhishek Rao, San Jose State University, California, U.S.A
Rohit Garg, San Jose State University, California, U.S.A
Mitesh Ambavale, San Jose State University, California, U.S.A
Raviteja Ainampudi, San Jose State University, California, U.S.A

pages: 30 - 35
Object Recognition Using Neural Networks and Complex Reflection Signals
Hristomir Yordanov, Technicl University Sofia, Bulgaria
Irina Topalova, University of Communication and Posts, Technicl University Sofia, Bulgaria

A Study on Round-trip Time Estimation from Unidirectional Packet Traces Using Different TCP Congestion Control Algorithms

Toshihiko Kato, Xiaofan Yan, Ryo Yamamoto, and Satoshi Ohzahata
Graduate School of Informatics and Engineering
University of Electro-Communications
Tokyo, Japan
e-mail: kato@is.uec.ac.jp, yanxiaofan@net.is.uec.ac.jp, ryo_yamamoto@is.uec.ac.jp, ohzahata@is.uec.ac.jp

Abstract—**Network operators often attempt to analyze traffic in the middle of their networks for various purposes. In such traffic analysis, the estimation of Round-Trip Time (RTT) is indispensable. Primarily, the RTT estimation is performed by consulting the relationship between a request and its response, such as a data segment and the associated ACK segment. However, in the middle of Internet, it is common that a network operator monitors traffic only in one direction. In such a case, an operator is required to estimate RTT from unidirectional packet traces. So far, several methods have been proposed for RTT estimation from unidirectional traces. Our previous paper showed a result that adopts the Lomb periodogram method to various TCP traces using different congestion control algorithms. In this paper, we show the RTT estimation using the autocorrelation based method and the Lomb periodogram method from unidirectional TCP traces, collected through Ethernet or wireless LAN, using different congestion control algorithms, i.e., TCP Reno, CUBIC TCP, TCP Vegas, and TCP Veno. As a result, the autocorrelation based method could not estimate RTT correctly, the Lomb periodogram method provided reasonable estimation, and the results by the Lomb periodogram method are not accurate enough for subtle analysis, such as congestion window estimation.**

Keywords- Unidirectional Packet trace; Round-trip Time; Autocorrelation; Lomb Periodogram; Congestion Control.

I. INTRODUCTION

This paper is an extension of our previous paper [1], which is presented in an IARIA conference.

Traffic analysis in the middle of Internet is an important issue for network operators. It can be applied the traffic classification, the traffic demand forecasting, and the malicious traffic detection. In the previous paper, we proposed a method to infer TCP congestion control algorithm from passively collected packet traces [2]. It adopts the following approaches.

(1) Focus on a specific TCP flow using source/destination IP addresses and ports.
(2) From the mapping between data segments and acknowledgment (ACK) segments, estimating Round-Trip Time (RTT) of the focused flow.
(3) Estimate a congestion window size (cwnd) from the data size transferred during one RTT.
(4) Obtain a sequence of cwnd values, and calculate a sequence of cwnd difference between adjacent cwnd values (we call ⊿cwnd).
(5) From the mapping between cwnd and ⊿cwnd, infer a congestion control algorithm for the TCP flow.

This method requires a bidirectional trace to obtain both data and ACK segments.

In actual networks, however, it is often possible that only unidirectional traces are collected in the middle of networks. In this case, the above method cannot be applied. So, in another previous paper, we tried to modify the above method to infer TCP congestion control algorithms from unidirectional traces [3]. In the modified method, a fixed time duration is used instead of RTT, and data size transferred during this duration was handled as cwnd. As a result, congestion control algorithms were estimated in some cases, but not in other cases. This is because our method depends largely on RTT value.

On the other hand, the estimation of RTT from traces has been actively studied and there are several proposals [4]-[7]. The RTT estimation methods proposed so far are classified into three categories. One is a method called Data-to-ACK-to-Data, which measures time between a data segment and the data segment sent just after the first data segment is ACKed [4]-[6]. This requires bidirectional packet traces and our first paper used it. Next is a method based on the autocorrelation [5][6]. This method counts the number of data segments in a short interval, and makes an array of counts indexed by the normalized interval. Then, it calculates the autocorrelation over the array and takes the maximum as a RTT. This method can be applied to unidirectional packet traces. The third one is use of spectral analysis [6][7]. A sequence of data segments are handled as a pulse function of time, which takes 1 when there is a data segment. Then, the frequency characteristic of this function is analyzed and the inverse of first harmonic is taken as RTT. Since the interval of data is irregular, the spectral analysis is performed by the Lomb periodogram [8].

In our previous paper [1], we picked up the third method for estimating RTT from unidirectional traces including different TCP congestion control algorithms, which we used for inferring congestion control algorithms [2][3]. In this paper, we add the results of RTT estimation by use of the second method, the autocorrelation based method, and discuss the results in more detail.

The rest of this paper is organized as follows. Section II explains the problems we suffered from in our previous work on estimating congestion window sizes from unidirectional packet traces [3]. Section III explains the conventional RTT estimation methods in detail. Section IV gives the results of RTT estimation for different TCP congestion control algorithms, using the autocorrelation based method and the Lomb periodogram method, and compare the results. In the end, Section V concludes this paper.

II. Related Work Problems on Congestion Window Size Estimation from Unidirectional Traces

A. *Problems on congestion window size estimation from unidirectional traces*

In our previous papers [2][3], we collected packet traces in the configuration shown in Figure 1. A TCP data sender is connected with a bridge through 100 Mbps Ethernet. The bridge inserts 100 msec RTT (50 msec delay for each direction) and 0.01% packet losses. The bridge is connected with a TCP data receiver through IEEE 11g wireless LAN (WLAN) or 100 Mbps Ethernet. The packet trace is collected at the TCP sender side. The collected traces include bidirectional ones, and in the unidirectional analysis, we picked up only data segments from the TCP sender to the TCP receiver.

Figures 2 and 3 show the results for CUBIC TCP [9] and TCP Vegas [10]. In the analysis a from bidirectional trace, cwnd and ⊿cwnd (both in bytes) are estimated in the way described in Section I, and their relationship is given in the figures (by blue dots). In the analysis from a unidirectional trace, we assumed that RTT is 100 msec. The data size transferred during 100 msec and its difference are called sentData and ⊿sentData (both in bytes), respectively, and shown in the figures by orange dots. In the case of CUBIC TCP, both results show the similar graph, which is a function in the form of $\left(\sqrt[3]{cnwd}\right)^2$ with decreasing and increasing parts [2]. This result means that the unidirectional analysis works well. In the case of TCP Vegas, however, the results for bidirectional analysis and unidirectional analysis are significantly different. According to the Vegas algorithm, ⊿cwnd takes 1,460 bytes (one segment size), 0, or -1,460 bytes independently of cwnd values, which is represented by the blue dots [2]. But, in the result for unidirectional analysis, the ⊿sentData values indicated by the orange dots are unstable. So, the unidirectional analysis does not work well.

In our experiment, the trace for CUBIC TCP is collected in the configuration that uses Ethernet between the bridge and the TCP receiver, and that for TCP Vegas is collected by use of WLAN. This is one of the reasons. Figure 4 shows examples of the time variation of TCP sequence number for CUBIC TCP and TCP Vegas. In the case of CUBIC TCP, data segments are transferred in groups and there are idle time periods without any data transmissions. Therefore, in the

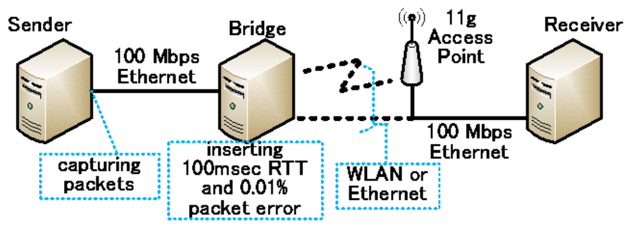


Figure 1. Experiment configuration.

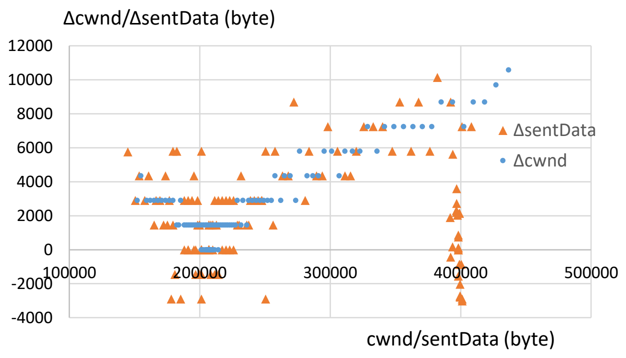


Figure 2. Result for CUBIC TCP [2][3].

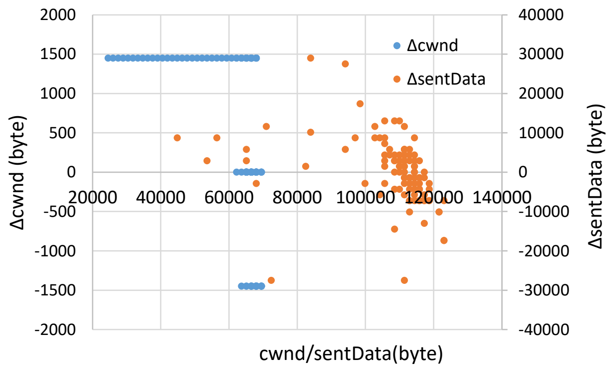


Figure 3. Result for TCP Vegas [2][3].

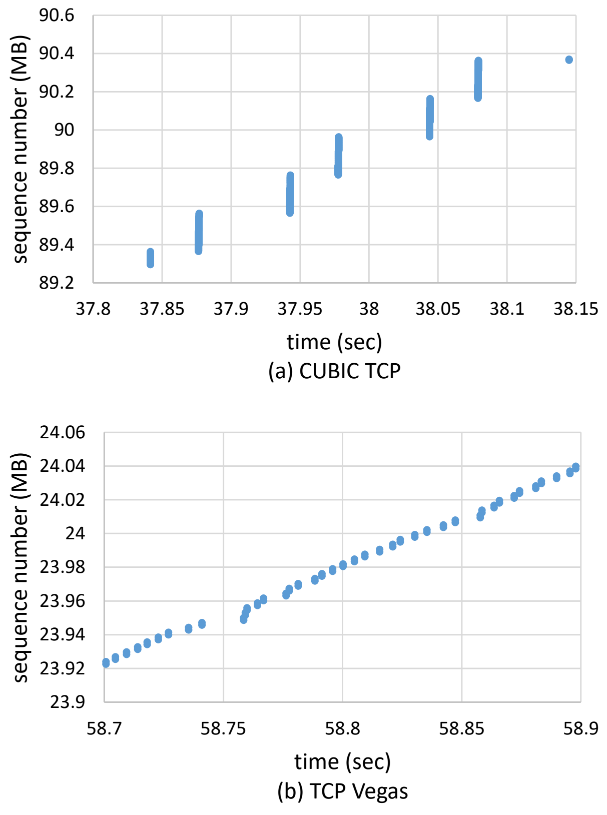


Figure 4. Sequence number vs. time.

unidirectional analysis, a sequence of data segments sent within a congestion window can be traced by use of 100 msec, which is a RTT determined tentatively. But, in the case of TCP Vegas, data segments are transmitted contiguously, and therefore, if RTT is not estimated correctly, a sentData value does not match the real cwnd value.

There considerations mean that the RTT estimation is critical for inferring TCP congestion control algorithms.

III. RTT ESTIMATION METHODS

As described in Section I, the RTT estimation methods are classified into three categories; the Data-to-ACK-to-Data method, the autocorrelation based method, and the Lomb periodogram method.

A. *Data-to-Ack-to-Data method*

The Data-to-ACK-to-Data method is illustrated in Figure 5. Since there is some transmission delay between a TCP data sender and a monitor capturing packet traces, the following procedure is used to estimate RTT between sender and receiver. (1) A monitor focuses on a data segment, and remembers the time (t1). (2) A monitor catches the ACK segment that acknowledges the data segment. (3) A monitor detects the data segment sent by the sender just after the ACK segment in (2), and remember the time (t2). (4) t3 – t1 is a RTT for this moment. In order to detect data segment (3), the TCP time stamp option is used.

B. *Autocorrelation based method*

In the autocorrelation based method, the RTT estimation is performed once per measurement interval *T*. An array $P[n]$ maintaining the count of data segments is prepared using unit time Δt, where *n* is ranging from 0 to ${}^{T}/_{\Delta t} - 1$. If a data segment is detected at an interval $[start\ time + m \cdot \Delta t$, $start\ time + (m+1) \cdot \Delta t)$, one is added to $P[m]$. For all the data segments from *start time* to *start time* +*T*, the array $P[n]$ is arranged. After that, the autocorrelation function is defined as

$$A(l) = \frac{1}{{}^{T}/_{\Delta t} - l} \sum_{j=0}^{{}^{T}/_{\Delta t} - l} P[j] \cdot P[j+l]. \qquad (1)$$

for lags $l = 0 \cdots {}^{T}/_{\Delta t} - 1$. RTT is computed as $\max(A)$. This method can be applied to the unidirectional analysis, and will work well for the cases that data segments are distributed unevenly in a trace, such as the case of CUBIC TCP in Figure 4(a).

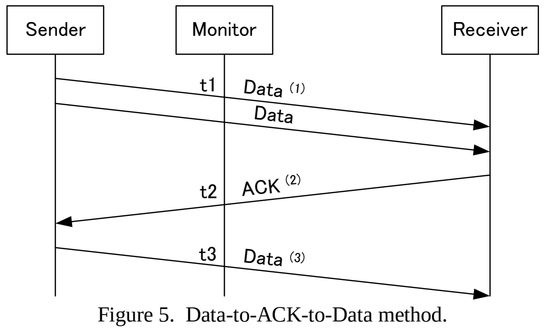


Figure 5. Data-to-ACK-to-Data method.

C. *Lomb periodogram method*

The last method is one based on the spectral analysis, in which a sequence of data segments are handled as a pulse time sequence, the frequency characteristic of this time sequence is analyzed, and the inverse of first harmonic is taken as RTT. Traditional spectral analysis, such as Fast Fourier Transform (FFT) assume that time domain data are regularly sampled [11]. However, in the RTT estimation, the time domain data is packet inter-arrival time of a specific flow. This data is sampled at each data packet capturing. This means that the time domain data in this case is irregularly sampled. In the case of the spectral analysis for irregularly sampled data, the Lomb periodogram is commonly used [8].

In the RTT estimation based on the Lomb periodogram, time sequence $\{t_i\}\ (i = 1, \cdots)$ is considered as an input, where t_i corresponds to one data segment capturing time. At a specific time t_k, the frequency characteristic of this time sequence is calculated using *N* time samples $t_{k-N+1}, \cdots t_k$ in the following way ($k > N$) [7].

- The minimum and maximum frequencies of the range for power spectrum are defined as
$$f_k^{min} = \frac{1}{t_k - t_{k-N+1}} \text{ and } f_k^{max} = \frac{N}{2} f_k^{min}.$$
Accordingly, the power spectrum is calculated for angular frequency
$$\omega_i = 2\pi f_k^{min} + i\Delta_\omega \ \ (i = 0 \cdots 2N-1),$$
where $\Delta_\omega = 2\pi \frac{f_k^{max} - f_k^{min}}{2N}$.
- The power spectrum at angular frequency ω_i is defined as
$$P_k^N(\omega_i) = \frac{1}{2\sigma_k^2} \left\{ \frac{\left[\sum_{j=0}^{N-1} (h_{k-j} - \bar{h}_k) cos\omega_i (t_{k-j} - \tau_k)\right]^2}{\sum_{j=0}^{N-1} cos^2 \omega_i (t_{k-j} - \tau_k)} + \frac{\left[\sum_{j=0}^{N-1} (h_{k-j} - \bar{h}_k) sin\omega_i (t_{k-j} - \tau_k)\right]^2}{\sum_{j=0}^{N-1} sin^2 \omega_i (t_{k-j} - \tau_k)} \right\} \qquad (2)$$
where $\bar{h}_k$ and σ_k^2 are the mean and variance of *N* samples of h_k:
$$\bar{h}_k = \frac{1}{N} \sum_{j=0}^{N-1} h_{k-j} \qquad (3)$$
$$\sigma_k^2 = \frac{1}{N-1} \sum_{j=0}^{N-1} h_{k-j}^2 - \frac{N}{N-1} \bar{h}_k^{\ 2}, \qquad (4)$$
and where τ_k is the solution of:
$$tan(2\omega_i \tau_k) = \frac{\sum_{j=0}^{N-1} sin2\omega_i t_{k-j}}{\sum_{j=0}^{N-1} cos2\omega_i t_{k-j}}. \qquad (5)$$

From the $2N - 1$power spectrum values specified in an $\omega - P(\omega)$ plane, local maximum values are calculated. Among the frequencies generating local maximum power spectrum values, the fundamental frequency f_0 is estimated under the condition that other frequencies generating local maximum values are multiples of f_0. At last, $T = {}^{1}/_{f_0}$ is the estimated RTT.

IV. RESULTS OF RTT ESTIMATION FOR VARIOUS CONGESTION CONTROL ALGORITHMS

This section describes the results of RTT estimation for various types of TCP traces with different congestion control algorithms. We use the packet traces used in our previous papers [2][3]. As described in Section II, these traces are collected at the sender side in the configuration shown in

Figure 1. Since packet losses are inserted at the bridge, we picked up a part of packet traces where no packet losses are detected, that is, where the sequence number of TCP segments keeps increasing. The traces themselves have bidirectional packet information and only the capturing time of data segments is extracted to build unidirectional traces. Together with the extraction, the real RTT is calculated from the mapping between data segments and ACK segments based on the Data-to-ACK-to-Data method.

A. Result for traces including TCP Reno

(1) Overview

TCP Reno is a classic congestion control method which adopts an Additive Increase and Multiplicative Decrease (AIMD) algorithm. Here, cwnd is increased each time the TCP sender receives an ACK segment acknowledging new data. The increase is $\frac{1}{cwnd}$ segments during the congestion avoidance phase, and as a result, cwnd is expected to be increased by one segment during one RTT.

The Reno packet trace we used here is collected in the network configuration with Ethernet (see Figure 1), and we picked up a part from 27.010458 sec to 45.99513 sec in the trace, where no retransmissions are detected for 7068 data segments.

(2) Results of autocorrelation based method

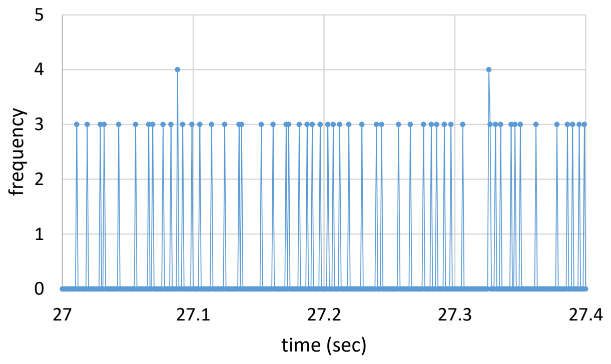
In the autocorrelation based method, we adopted 1msec as the unit time (Δt). This means that we can estimate RTT in the order of mili seconds, which will be enough for discussing the estimation capability of this method. We adopted 400 msec as the measurement interval (T). By use of these values, the autocorrelation can be calculated with changing the time lag from 0 to 399. We picked up the autocorrelation in the range of $l = 1 \cdots 200$.

Figure 6 shows the results of the applying autocorrelation based method to the TCP Reno traced mentioned above. Figure 6(a) shows the histogram result between 27.0 sec and 27.4 sec. Three or four data segments are transferred within 1 msec interval. The intervals with data segments are repeated in a duration of a few msec and 20 msec. Figure 6(b) shows the result of autocorrelation for the data shown in Figure 6(a). The range of time lag is 0 through 200, but we use the range of $l = 1 \cdots 200$. From this result, we can select 105 as the lag value which generates the largest autocorrelation. So, we estimated that the RTT at time 27.0 sec is 105 msec. Similarly, we estimated RTT for every 400 msec from the obtained packet trace. Figure 6(c) shows the results. This figure also shows the actual RTT (indicated as RTT in the figure), the RTT estimated by the Data-to-ACK-to-Data method by use of bidirectional information contained in the original packet trace. The actual RTT is stable at 100 msec, but the estimated RTT changes between 0 msec and 200 msec although 60% of the results are close to 100 msec.

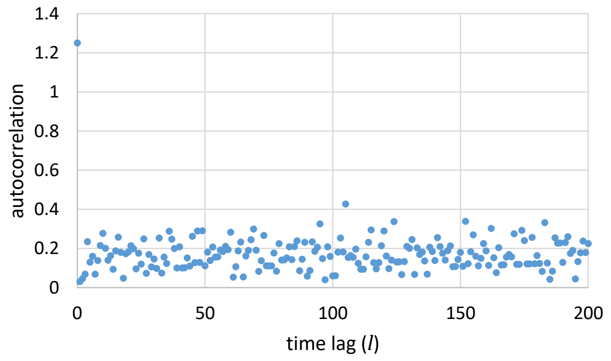
(3) Results of Lomb periodogram method

In order to apply the Lomb periodogram method, we need to decide the value of N. We used $N = 500$ in calculating the Lomb periodogram. Figure 7 shows a result of RTT estimation from the Reno trace. Figure 7(a) is the result for periodogram at time 28.156143 sec. The horizontal axis is an angular frequency and the vertical axis is a periodogram. This figure shows there are several peaks periodically. Figure 7(b) zooms up the low angular frequency part of Figure 7(a). It shows that there are harmonized frequencies such that there are large periodogram values at some frequencies which are integral multiple of a specific frequency (fundamental frequency f_0). In Figure 7(b), angular frequencies 61.4343, 118.7525, and 181.5296 are those frequencies. From this result, we can conclude that $2\pi f_0 = 61.4343$. So, we obtain $f_0 = 9.77755$ and RTT $= {}^{1}/_{f_0} = 0.102275$sec.

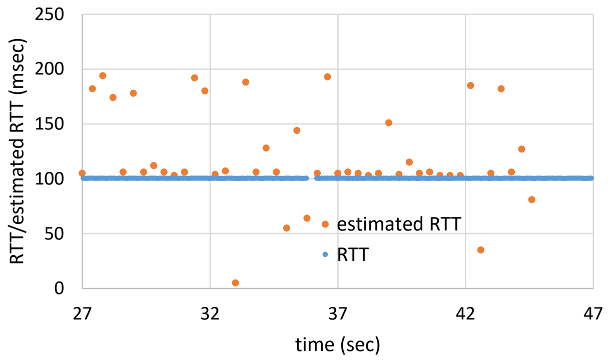
We conducted similar calculations for multiple points of time in the trace and obtained the estimated RTT as shown in



(a) histogram of data segments between 27 sec and 27.4 sec



(b) autocorrelation for interval [27.0, 27.4]

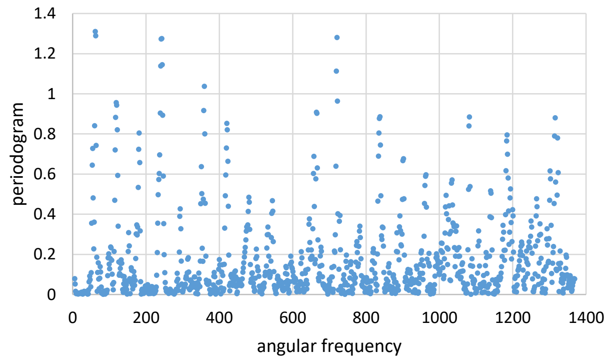


(c) estimated RTT and actual RTT

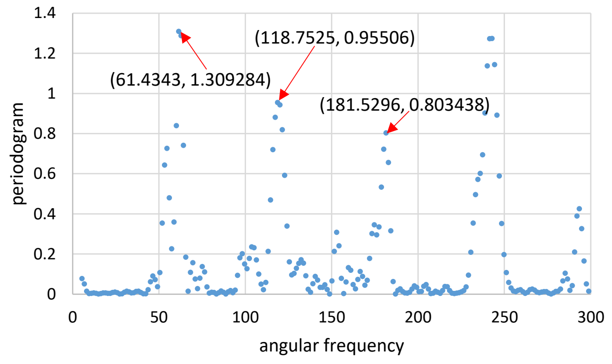
Figure 6. RTT estimation from Reno trace by autocorrelation.

Figure 7(c). This figure also gives actual RTT values obtained from the relationship data and ACK segments in the original trace information. This result says that, although the actual RTT is extremely stable at 100 msec, the estimated RTT includes some errors in the order of 10 msec. When this result is compared with the result by the autocorrelation based method, that by the Lomb periodogram method is better than that by the autocorrelation based method.

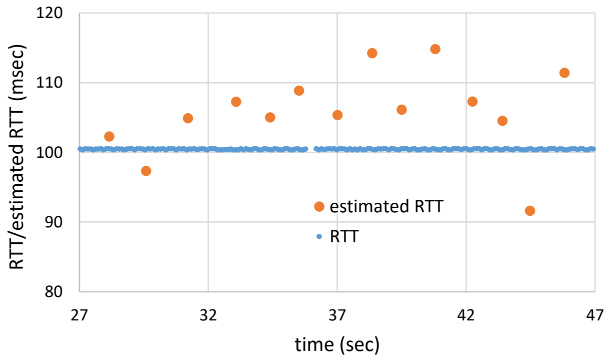
The reason that the actual RTT is stable is that this experiment is conducted through only Ethernet and that there are no large delay variations. However, the RTT estimation by use of neither the autocorrelation nor the Lomb periodogram can reflect this situation.



(a) periodogram at time 27.03713 sec



(b) zooming up low angular frequency part



(c) estimated RTT and actual RTT

Figure 7. RTT estimation from Reno trace by Lomb periodogram.

B. *Result for traces including CUBIC TCP*

(1) Overview

As described in Section II, CUBIC TCP defines *cwnd* as a cubic function of elapsed time T since the last congestion event [9]. Specifically, it defines cwnd by (6).

$$cwnd = C\left(T - \sqrt[3]{\beta \cdot \frac{cwnd_{max}}{C}}\right)^3 + cwnd_{max} \qquad (6)$$

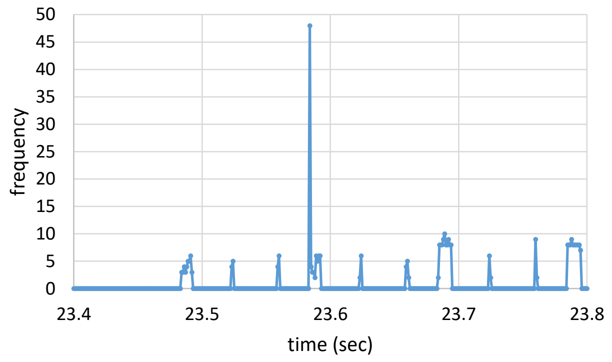
Here, C is a predefined constant, β is the decrease parameter, and $cwnd_{max}$ is the value of *cwnd* just before the loss detection in the last congestion event. Comparing with TCP Reno, cwnd increases faster in CUBIC TCP.

We estimated RTT from the unidirectional packet trace including only data segments with CUBIC TCP. The trace is collected in the configuration using only Ethernet. We picked up a part in the trace from 23.483123 sec. to 38.348383 sec. for the RTT estimation.

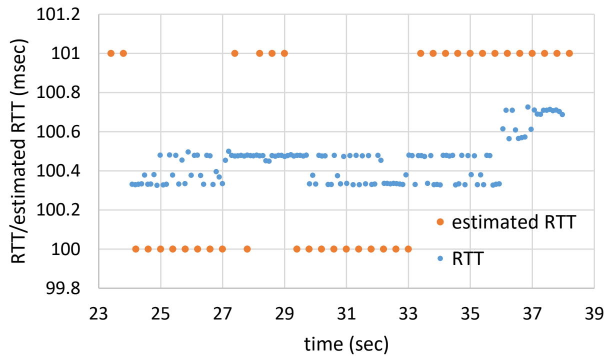
(2) Results of autocorrelation based method

For the autocorrelation based method, we used the same parameters as those for TCP Reno. That is, 1msec is as Δt, 400 msec is as T, and the autocorrelation is evaluated in the range of l is from 1 to 200.

The results are shown in Figure 8. Figure 8(a) shows the histogram result between 23.4 sec and 23.8 sec. Compared with the case of TCP Reno given in Figure 6(a), data segments are transmitted in a group in the case of CUBIC TCP. A



(a) histogram of data segments between 23.4 sec and 23.8 sec



(b) estimated RTT and actual RTT

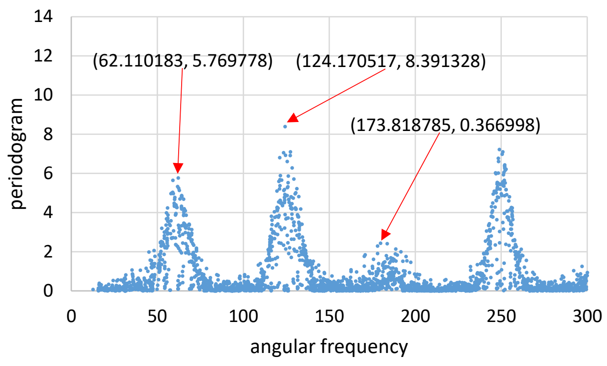
Figure 8. RTT estimation from CUBIC trace by autocorrelation.

portion where data segments are sent over multiple time slots is repeated every one hundred mili second, and so it is considered that the RTT can be easily estimated by the autocorrelation. Figure 8(b) shows the estimated RTT, together with the actual RTT. The estimated RTT takes either 100 msec or 101 msec, and the actual RTT takes values between them. Since the granularity of the estimated RTT is 1 msec, it can be said that the autocorrelation based method provides good estimation for CUBIC TCP.

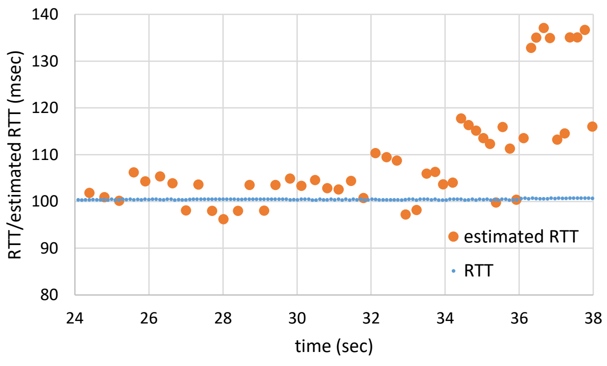
(3) Results of Lomb periodogram method

By applying the Lomb periodogram similarly with the case of Reno, we obtained estimated RTT as shown in Figure 8. Figure 9(a) shows the result for periodogram at time 23.987461 sec. In this figure, there are peaks of periodogram at angular frequencies of 62.110183, 124.170517, and so on. So, we estimated that the fundamental angular frequency is 62.110183, and calculated the estimated RTT at this timing.

Figure 9(b) shows the estimated RTT together with the actual RTT values. The results show that the actual RTT is stable at 100 msec and, on the other hand, the estimated RTT changes a lot between 90 msec and 140 msec. The fluctuation is larger for CUBIC than TCP Reno. Especially, the difference between the estimated RTT and the actual RTT becomes large when the time is between 36 sec and 38 sec. During this period, the cwnd value itself becomes large and the large cwnd value may give some bad influence to the RTT estimation.



(a) periodogram at time 23.987461 sec



(b) estimated RTT and actual RTT

Figure 9. RTT estimation from CUBIC trace by Lomb periodogram.

In the case of CUBIC TCP, the autocorrelation based method could provide more precise RTT estimation than the Lomb periodogram method. The reason is considered to be that the burstiness of the logged packet sequence is high in this case.

C. Result for traces including TCP Vegas

(1) Overview

TCP Vegas estimates the bottleneck buffer size using the current values of *cwnd* and RTT, and the minimal RTT for the TCP connection, according to (7) [10].

$$BufferSize = cwnd \times \frac{RTT - RTT_{min}}{RTT} \qquad (7)$$

At every RTT interval, Vegas uses this *BufferSize* to control *cwnd* in the congestion avoidance phase in the following way.

$$\varDelta cwnd = \begin{cases} 1 & (BufferSize < A) \\ 0 & (A \leqq BufferSize \leqq B) \\ -1 & (BufferSize > B) \end{cases} \qquad (8)$$

Here, A = 2 and B = 4 (in unit of segment) are used in the Linux operating system.

We estimated RTT from the unidirectional packet trace including only data segments with TCP Vegas. In this case, in contrast with the above cases, the trace is collected in the configuration using WLAN. We picked up a part in the trace from 37.988347 sec to 59.699611 sec for the RTT estimation.

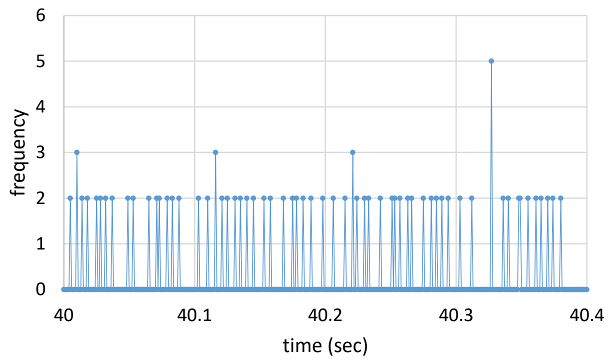
(2) Results of autocorrelation based method

Figure 10 shows the results of the RTT estimation by use of the same parameters used in the cases of TCP Reno and CUBIC TCP. Figure 10(a) shows the histogram result between 40.0 sec and 40.4 sec. In this case, the frequency is almost two packets and the timing when some packets are transmitted is scattered. That is, the packet transmission is not bursty for the Vegas packet trace used in this experiment. Figure 10(b) shows the estimated RTT and the actual RTT. As supposed from the result of the histogram, the estimated RTT is distributed between 0 mse and 200 msec. Since 200 msec is the upper bound in the estimation, it is said that the estimation here is not done well but the estimated RTT is randomly distributed.

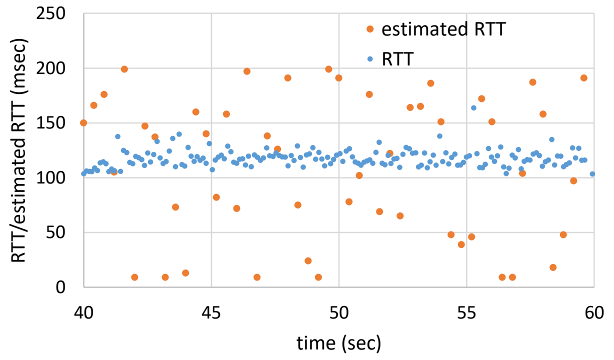
(3) Results of Lomb periodogram method

Figure 11 shows the result of the RTT estimation based on the Lomb periodogram method. Figure 11(a) is the periodogram at time 40.082778 sec. From this result, we obtain that the fundamental angular frequency is 58.288021 and the estimated RTT at this point is 107.795 msec.

Figure 11(b) shows the estimated RTT and actual RTT obtained for the part of the Vegas packet trace mentioned above. In this case, the estimated RTT is stable around 100 msec, and on the other hand, the actual RTT values are scattered between 100 msec and 140 msec. That is, although the actual RTT is changing, the RTT estimated by the Lomb periodogram does not follow the fluctuation. As we indicated in Figure 4(b) and Figure 10(a), the timing of capturing data segments is almost uniformly distributed in this case. As a

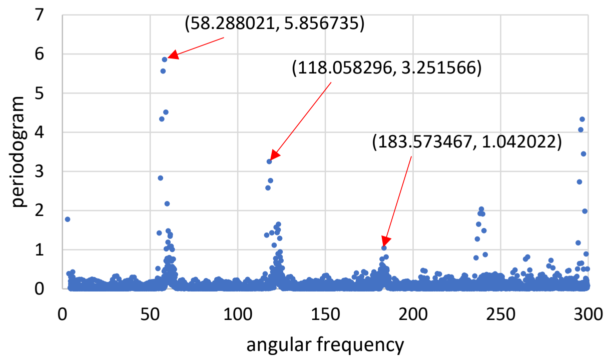


(a) histogram of data segments between 40.0 sec and 40.4 sec

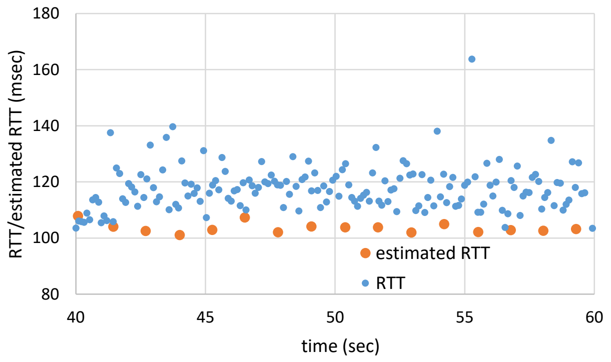


(b) estimated RTT and actual RTT

Figure 10. RTT estimation from Vegas trace by autocorrelation.



(a) periodogram at time 40.082778 sec



(b) estimated RTT and actual RTT

Figure 11. RTT estimation from Vegas trace by Lomb periodogram.

result, it is considered that the Lomb periodogram method cannot detect the actual RTT. Compared with the result of the autocorrelation based method, however, the Lomb periodogram method provides much better estimation in this case.

D. Result for traces including TCP Veno

(1) Overview

TCP Veno (Vegas and ReNO) [12] is an example of hybrid type congestion control method, considering packet losses and delay. It uses the *BufferSize* in (7) to adjust the growth of *cwnd* in the congestion avoidance phase as follows. If $BufferSize > B$ (B is the Vegas parameter B), cwnd grows by 1/cwnd for every other new ACK segment, and otherwise, it grows in the same manner with TCP Reno. That is, when the congestion status is heavy, i.e., the bottleneck buffer size is large, the increasing rate of cwnd is halved.

We estimated RTT from the unidirectional Veno trace captured in the WLAN configuration in Figure 1. We picked up a part in the trace from 37.684643 sec to 52.653736 sec including 23,360 data segments.

(2) Results of autocorrelation based method

Figure 12 shows the results of the RTT estimation by the autocorrelation for TCP Veno. Figure 12(a) is the histogram result between 38.0 sec and 38.4 sec. Similarly with the result for TCP Vegas shown in Figure 10(a), the data transmissions are scattered, that is, not bursty. The frequency of time slots with data transmission is two or three, and the intervals between those time slots are less than 20 msec. As a result, the estimated RTT values shown in Figure 12(b) are largely different from the actual RTT values. Especially, in the time frame later than 41.4 sec, the estimated RTT is 1 msec. In the Veno packet trace used here, the autocorrelation based method was very poor in the RTT estimation.

(3) Results of Lomb periodogram method

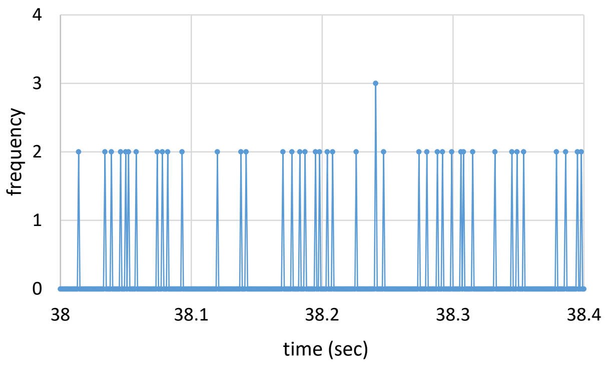
Figure 13 shows the results of the RTT estimation by the Lomb periodogram for TCP Veno. Figure 13(a) is the periodogram at time 38.090911 sec. This result indicates that the fundamental angular frequency is 63.281391 and that the estimated RTT at this point is 99.239 msec.

Figure 13(b) shows the estimated RTT and the actual RTT for TCP Veno. Similarly with the results for TCP Vegas, the estimated RTT values are stable around 100 msec, but the actual RTT has a distribution between 100 msec and 130 msec. In this sense, the Lomb periodogram method cannot estimate RTT in a strict sense, but it provides a reasonable estimation compared with the autocorrelation based method.

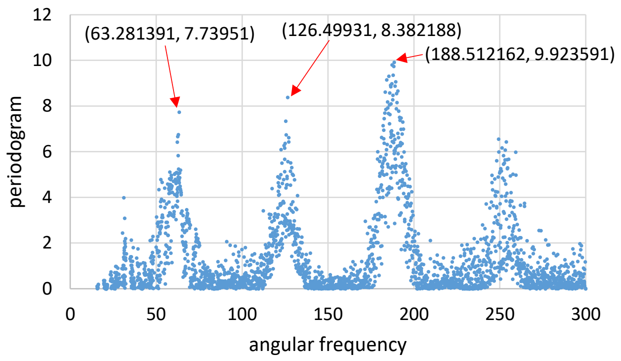
E. Discussions

Through the experiments described above, we obtained the following discussions.

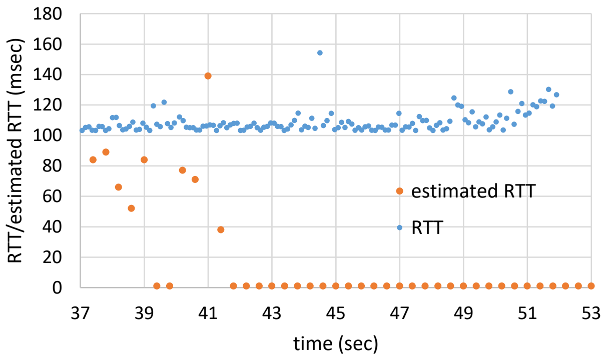
- First of all, the autocorrelation based method could not estimate RTT correctly in many cases. In our experiment, three cases out of four did not work well. In the case of



(a) histogram of data segments between 38.0 sec and 38.4 sec

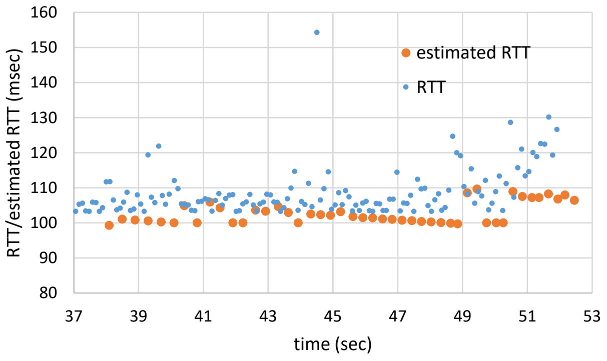


(a) periodogram at time 38.090911 sec



(b) estimated RTT and actual RTT

Figure 12. RTT estimation from Veno trace by autocorrelation.



(b) estimated RTT and actual RTT

Figure 13. RTT estimation from VENO trace by Lomb periodogram.

CUBIC TCP, this method could estimate RTT accurately. The reason is considered that data segments in unidirectional packet traces are transmitted in a burst and that the actual RTT is stable. In other cases, i.e., for TCP Reno, TCP Vegas, and TCP Veno, the data transmission is performed relatively in a uniform way, and this situation makes the RTT estimation difficult by the autocorrelation based method.

- Secondly, compared with the autocorrelation method, the Lomb periodogram method was possible to estimate approximate RTT values from unidirectional packet traces. Strictly speaking, however, the estimated RTT values have some errors and they are not tolerable for the approaches that require accurate RTT estimation, such as our method to infer the TCP congestion algorithms from unidirectional packet traces [3]. Moreover, although the experiments adopted here added a fix delay, actual TCP communications suffer from variable delay like Bufferbloat [13]. So, the accurate estimation will be more difficult in real environments.

- The third point is that the estimation by the Lomb periodogram method is affected largely by the network configuration, such as with Ethernet or with WLAN. It is also affected somehow by the congestion control used in packet traces. In our experiment, the traces of TCP Reno and CUBIC TCP were collected in an Ethernet configuration. In this case, the actual RTT was stable and the estimated RTT was fluctuated. In the CUBIC TCP trace, where the congestion control is more aggressive, the errors of the estimated RTT increased. On the other hand, the traces of TCP Vegas and TCP Veno were collected in a WLAN configuration. In this case, while the actual RTT was fluctuated, the Lomb periodogram method could not estimate this fluctuation and the estimated RTT was stable.

V. CONCLUSIONS

This paper described the results of applying the autocorrelation based method and the Lomb periodogram method to estimating RTT from unidirectional packet traces including TCP segments with different congestion control algorithms, TCP Reno, CUBIC TCP, TCP Vegas, and TCP Veno. The performance evaluation gave the following results.

First of all, the autocorrelation based method provided poor performance in the RTT estimation. Only CUBIC TCP in our experiment worked well, and the estimation for other three congestion control algorithms was not successful.

Secondly, the Lomb periodogram method was possible to estimate more accurate RTT values than the autocorrelation based method. However, the estimated RTT values were not accurate enough for the applications requiring precise RTT estimation, such as our method to infer the TCP congestion algorithms [3].

Lastly, we confirmed a tendency that the estimation by the Lomb periodogram is affected by the network configuration, such as with Ethernet or with WLAN. In the Ethernet configuration, the actual RTT is stable but the estimated RTT is fluctuated. In the WLAN configuration, the result is opposite.

In conclusion, it will be considered that the accurate RTT estimation will be difficult from unidirectional packet traces, although the rough estimation will be feasible by the Lomb periodogram method.

REFERENCES

[1] T. Kato, X. Yan, R. Yamamoto, and S. Ohzahata, "Applying Lomb Periodogram to Round-trip Time Estimation from Unidirectional Packet Traces with Different TCP Congestion Controls," IARIA ICIMP 2018, pp. 1-6, Jul. 2018.

[2] T. Kato, A. Oda, C. Wu, and S. Ohzahata, "Comparing TCP Congestion Control Algorithms Based on Passively Collected Packet Traces," IARIA ICSNC 2015, pp. 135-141, Nov. 2015.

[3] T. Kato, L. Yongxialee, R. Yamamoto, and S. Ohzahata, "How to Characterize TCP Congestion Control Algorithms from Unidirectional Packet Traces," IARIA ICIMP 2016, pp. 23-28, May 2016.

[4] H. Jiang and C. Dovrolis, "Passive Estimation of TCP Round-Trip Times," ACM SIGCOMM Comp. Commun. Rev. vol. 32, issue 3, pp. 75-88, Jul. 2002.

[5] B. Veal, K. Li, and D. Lowenthal, "New Methods for Passive Estimation of TCP Round-Trip Times," Passive and Active Nework Measurement, PAM 2005, LNCS, vol. 3431, pp. 121-134.

[6] R. Lance and I. Frommer, "Round-Trip Time Inference Via Pasive Monitoring," ACM SIGMETRICS Perf. Eval. Rev., vol. 33, issue 3, pp. 32-38, Dec. 2005.

[7] D. Carra et al., "Passive Online RTT Estimation for Flow-Aware Routers Using One-Way Traffic," NETWORKING 2010 LNCS6091, pp. 109-121, 2010.

[8] J. Scargle, "Statistical aspects of spacial analysis of unevently spaced data," J. Astrophysics, vol 263, pp. 835-853, Dec. 1982.

[9] S. Ha, I. Rhee, and L. Xu, "CUBIC: A New TCP-Friendly High-Speed TCP Variant," ACM SIGOPS Op. Syst. Review, vol. 42, issue 5, pp. 64-74, Jul. 2008.

[10] L. Brakmo and L. Peterson, "TCP Vegas: End to End Congestion Avoidance on a Global Internet," IEEE J. Sel. Areas Commun., vol. 13, no. 8, pp. 1465-1480, Oct. 1995.

[11] S. Kay and S. Marple, "Spectrum analysis; A modern perspective," Proc. of the IEEE, vol. 69, issue 11, pp. 1380-1419, Nov. 1981.

[12] C Fu and C. Liew, "TCP Veno: TCP enhancement for transmission over wireless access networks," IEEE J. Sel. Areas Commun., vol. 21, no. 2, Feb. 2003.

[13] S. Strowes, "Passively Measuring TCP Round-trip Times," ACM Queue, vol. 11, issue 8, pp. 1-12, Aug. 2013.

Experimental Modeling and Measurements of Networked Wireless Sensors for Power Consumption

Jin-Shyan Lee
Department of Electrical Engineering
National Taipei University of Technology
Taipei, Taiwan
e-mail: jslee@mail.ntut.edu.tw

Jorge Portilla, Gabriel Mujica
Centro de Electronica Industrial
Universidad Politecnica de Madrid
Madrid, Spain
email: jorge.portilla@upm.es
email: gabriel.mujica@upm.es

Yuan-Heng Sun
Information and Communications Labs
Industrial Technology Research Institute
Hsinchu, Taiwan
e-mail: gilbertsun@itri.org.tw

Abstract— **Power efficiency is a critical design issue in wireless sensor networks. In order to analyze the power consumption of a single node, a system model of networked wireless sensors is thus required. Based on a Petri net framework, this paper has preliminarily applied a systematic approach to the modeling and measurement of power consumption for ZigBee-equipped sensors. Moreover, several experiments have been conducted to measure the real power consumption of wireless sensor networks. It is believed that the experimental measurements presented in this paper would benefit application engineers in analyzing and understanding the power consumption of wireless sensor networks.**

Keywords-energy consumption; experimental measurements; Petri nets; sensor models; wireless sensor networks; ZigBee.

I. Introduction

An earlier version of this paper was presented at the International Conference on Systems and Networks Communications (ICSNC) and was published in its proceedings [1]. This paper extends the previous work by conducting more experiments regarding the power consumption of wireless sensor networks.

Recently, there has been an increasing emphasis on developing distributed Wireless Sensor Networks (WSNs) with self-organization capabilities to cope with device failures, changing environmental conditions, and different sensing and measurement applications [2]-[5]. WSNs consist of hundreds or even thousands of networked wireless sensors, which are linked by radio frequencies to perform distributed sensing tasks. In general, since these wireless sensors are equipped with batteries, energy consumption is a major design issue. Researchers have attempted to determine the best topology, the optimal way of routing, or whether the sensor node should aggregate data or not. All these topics are investigated with the intention of prolonging network lifetime from a global networking point of view [6]-[8].

On the other hand, from a single node point of view, energy conservation could be achieved by applying some power management techniques. However, in order to propose methods, by which power consumption can be minimized in networked wireless sensors, it is first necessary to gain an accurate understanding of their energy consumption characteristics. Thus, a system model of wireless sensors is required so as to analyze the energy consumption of a single node.

Starting from measurements carried out on the off-the-shelf radio, Bougard *et al.* [9] evaluated the potential of an IEEE 802.15.4 radio for use in an ultra-low power sensor node operating in a dense network. Their resulting model has been used to optimize the parameters of both the physical and medium access control layers in a dense sensor network scenario. Also, based on the empirical energy consumption measurements of Bluetooth modules, Ekstrom *et al.* [10] presented a realistic model of the radio energy consumption for Bluetooth-equipped sensor nodes used in a low-duty-cycle network. Their model gives users the possibility to optimize their radio communication with respect to energy consumption while sustaining the data rate. From a hybrid system point of view, Sousa *et al.* [11] modeled and analyzed the power consumption of a wireless sensor node in sensor networks using differential hybrid Petri Nets (PNs). With the discrete event evolution, the continuous battery discharge profile is updated and the remaining battery capacity is estimated. Moreover, their Petri net model was further applied to the design and evaluation of several dynamic power management solutions [12]. Based on Petri nets, Shareef and Zhu [13] also developed a model of a wireless sensor node that can accurately estimate the energy consumption. They used this model to identify an optimal threshold for powering down a sensor node of a specific wireless sensor application.

Most of the previous work focused on developing a conceptual sensor model and provided limited results on realistic measurement or comparative experiments. By applying our previously proposed Petri net framework in [14], this work has preliminarily modeled the energy consumption of a ZigBee-equipped sensor node. Then, a basic experiment has been conducted to measure the real power consumption and provide input parameters to the PN model, which could be applied to further simulations of ZigBee-based WSNs.

This is the sense to use the PN model to describe the power consumption of sensor nodes. Furthermore, by using the developed modular Cookie platforms [15]-[16], more experiments regarding measurements of power consumption for wireless sensor networks have conducted to analyze different power profiles to reach ultra-low-power states. It is believed that the experimental measurements presented in this paper would benefit application engineers in analyzing and understanding the power consumption of wireless sensor networks.

The rest of this paper is organized as follows. Section II introduces the Petri net modeling of wireless sensors. Next, preliminary experiments are provided in Section III. Then, Section IV gives more experimental measurements of power consumption for wireless sensor networks. Finally, Section V concludes this paper.

II. Petri Net Modeling of Wireless Sensors

This section will introduce the basic PN concepts, the typical PN modeling, the MultiParadigm Modeling (MPaM) methodology, and then illustrate the behavior modeling of networked wireless sensors.

A. *Basic PN Concepts* [5]

A PN is identified as a particular kind of bipartite directed graph populated by three types of objects. They are places, transitions, and directed arcs connecting places and transitions. Formally, a PN can be defined as

$$G = (P, T, I, O) \tag{1}$$

where,

$P = \{p_1, p_2, ..., p_m\}$ is a finite set of places, where $m > 0$;

$T = \{t_1, t_2, ..., t_n\}$ is a finite set of transitions with $P \cup T \neq \emptyset$ and, where $n > 0$;

$I : P \times T \to N$ is an input function that defines a set of directed arcs from *P* to *T*, where $N = \{0, 1, 2, \ldots\}$;

$O : T \times P \to N$ is an output function that defines a set of directed arcs from *T* to *P*;

A marked PN is denoted as (G, M_0), where M_0: $P \to N$ is the initial marking. A transition *t* is enabled if each input place *p* of *t* contains at least the number of tokens equal to the weight of the directed arc connecting *p* to *t*. When an enabled transition fires, it removes the tokens from its input places and deposits them on its output places. PN models are suitable to represent the systems that exhibit concurrency, conflict, and synchronization.

Several important PN properties include boundness, which means no capacity overflow, liveness, which shows the freedom from deadlock, conservativeness, which indicates the conservation of non-consumable resources, and reversibility, which represents the cyclic behavior.

The concept of liveness is closely related to the complete absence of deadlocks. A PN is said to be live if, no matter what marking has been reached from the initial marking, it is possible to ultimately fire any transition of the net by progressing through some further firing sequences. This means that a live PN guarantees deadlock-free operation, no matter what firing sequence is chosen. Validation methods of these properties include reachability analysis, invariant analysis, reduction method, siphons/traps-based approach, and simulation [17].

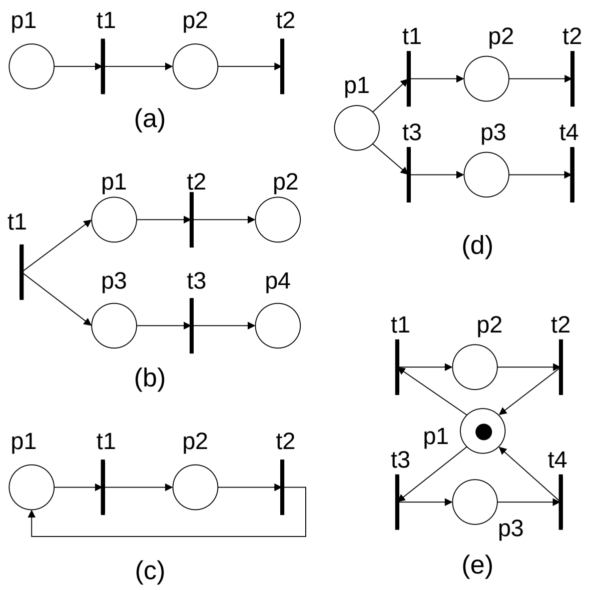


Figure 1. Basic PN models for (a) sequential, (b) concurrent, (c) cyclic, (d) conflicting, and (e) mutually exclusive relations [18].

B. *Typical PN Modeling* [18]

At the modeling stage, one needs to focus on the major operations and their sequential or precedent, concurrent, or conflicting relationships. The basic relations among these processes or operations can be classified as follows.

- *Sequential:* As shown in Figure 1 (a), if one operation follows the other, then the places and transitions representing them should form a cascade or sequential relation in PNs.
- *Concurrent:* If two or more operations are initiated by an event, they form a parallel structure starting with a transition, i.e., two or more places are the outputs of the same transition. An example is shown in Figure 1 (b). The pipeline concurrent operations can be represented with a sequentially-connected series of places/transitions, in which multiple places can be marked simultaneously or multiple transitions are enabled at certain markings.
- *Cyclic:* As shown in Figure 1 (c), if a sequence of operations follow one after another and the

completion of the last one initiates the first one, then a cyclic structure is formed among these operations.

- *Conflicting:* As shown in Figure 1 (d), if either of two or more operations can follow an operation, then two or more transitions create the outputs from the same place.
- *Mutually Exclusive:* As shown in Figure 1 (e), two processes are mutually exclusive if they cannot be performed at the same time due to constraints on the use of shared resources. A structure to realize this is through a joint place marked with one token plus multiple output and input arcs to activate these processes.

C. *MultiParadigm Modeling (MPaM)* [14]

To deal with specific and complicated problems, we have to integrate heterogeneous modeling arts, thereby resulting in the MPaM methodology. It is based on a proposition of giving different entities of a complex system the most appropriate modeling abstractions [14]. From a viewpoint of MPaM, the PN is adopted to design and analyze coordination controllers in a discrete-event domain. The primary motivation for employing PN as hybrid models is the situation that all those good characteristics that make discrete PN a valuable discrete-event model still be available to hybrid systems. Examples of these characteristics include: PN does not need the exhaustive enumeration of the state space at the design stage and can finitely model systems with an infinite state space. Moreover, PN provides a modular description where the structure of each module is maintained in the composed model. Furthermore, discrete states of PN are modeled by a vector and not by a symbolic label, thus linear algebraic techniques may be adopted for system analysis.

Figure 2 represents the previously proposed PN framework for modeling a system in discrete-event and discrete-time domains [14]. Each operation is modeled with a *command* transition to start the operation, a progressive *working* place, a *response* transition to end the operation, and a *completed* place. Note that the start transition (drawn with a dark symbol) is a controllable event as "command" input, while the end transition is an uncontrollable event as "response" output. The working place is a Hierarchical Hybrid Place (HHP, drawn with a triple circle), in which the state equations of the systems to be controlled are contained and interacted through the boundary interface. The interaction between event-driven and time-driven domains is realized in the following way: a token put into the working place triggers a discrete (or continuous) time process with the corresponding equations. Thresholds are monitored concurrently. Each threshold is corresponding to a transition, that is, the response transitions. When the threshold is reached or crossed, it indicates that the associated event is happening, and the corresponding transition is fired. Next, a new marking is evaluated, and the combination of the hybrid system restarts.

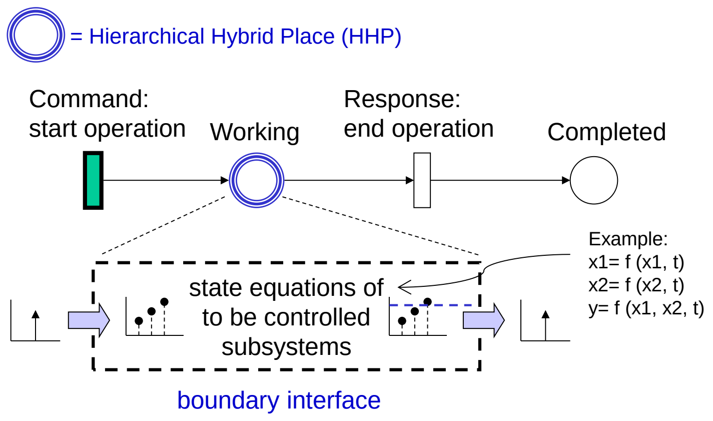


Figure 2. Multiparadigm modeling within a Petri net framework [14].

D. *Behavior Modeling of Networked Wireless Sensors*

In general, radio communication is the most energy consuming part of a wireless sensor as compared with its sensing and computation tasks. Hence, our model focuses on the operations of packet transmission and reception. By applying the design procedure in [14], the PN model of a networked wireless sensor is constructed as shown in Figure 3, which consists of 17 places and 14 transitions, respectively. The corresponding notations are described in Table I. The model is based on a scenario where a sensor node periodically transmits and receives some data towards, for example, a base station.

TABLE I
NOTATION FOR PETRI NET OF A WIRELESS SENSOR IN FIGURE 3

Place	Description	Transition	Description
p1	Node in sleep mode	t1	Cmd: start startup sequence
p2	MCU running at 16MHz	t2	Re: end startup sequence
p3	Startup sequence completed	t3	Cmd: start running MCU at 32MHz
p4	MCU running at 32MHz	t4	Re: end running MCU
p5	MCU running completed	t5	Cmd: start CSMA/CA operation
p6	Radio in RX mode	t6	Re: end CSMA/CA operation
p7	CSMA/CA operation completed	t7	Cmd: start transmitting packets
p8	Radio in TX mode	t8	Re: end transmitting packets
p9	Packet transmission completed	t9	Cmd: start receiving packets
p10	Radio in RX mode	t10	Re: end receiving packets
p11	Packet reception completed	t11	Cmd: start processing packets
p12	Processing packets	t12	Re: end processing packets
p13	Processing packets completed	t13	Cmd: start shutdown sequence
p14	MCU running at 16MHz	t14	Re: end shutdown sequence
p15	Shutdown sequence completed		
p16	MCU is available		
p17	Radio is available		

III. PRELIMINARY EXPERIMENTS

This section will firstly show the measurement setup and experimental results. Then, the comparisons between the measurement and PN model will be described.

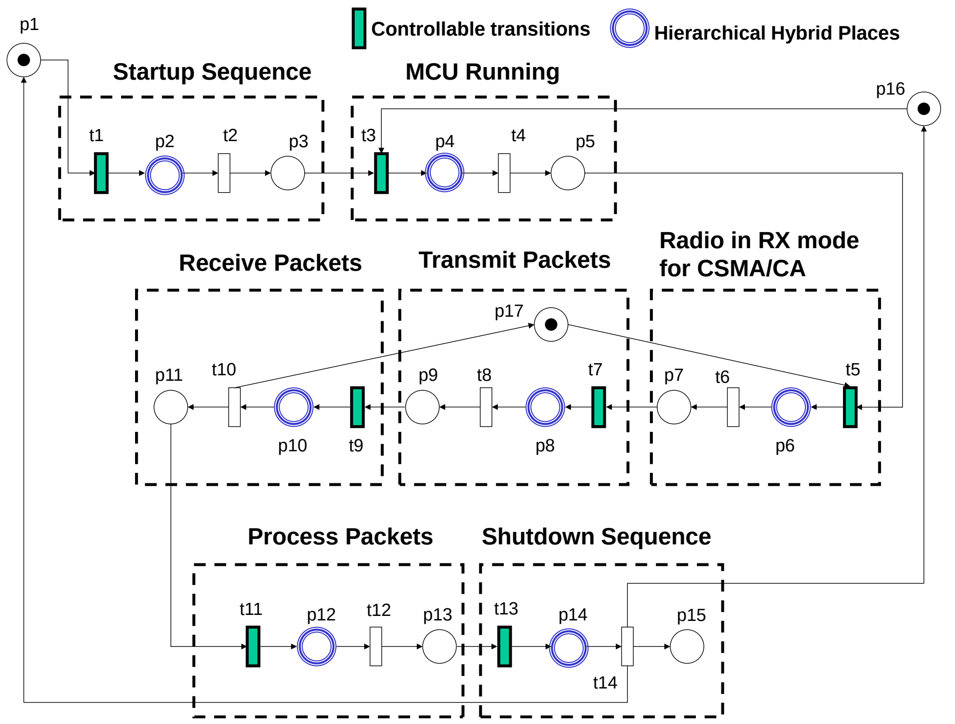


Figure 3. Petri net model of a networked wireless sensor.

A. *Measurement Setup*

In this section, the energy consumption of a wireless sensor as computed via its Petri net model will be compared against real measurements collected from a ZigBee-equipped sensor node. The measurement setup in [19] has been adopted as shown in Figure 4, in which a ZigBee End Device is the Device Under Test (DUT) and powered by a power supply. The energy consumption measurements are performed at the End Device, which periodically (every 0.5 sec in our measurement) wakes up and sends data to the coordinator (base station). The voltage across a 10 Ohm resistor is monitored to determine the current draw of the system. The measurement system has been calibrated with both a digital oscilloscope and a digital multimeter to ensure an accurate measurement. Figure 5 shows the hardware setup during energy consumption measurement.

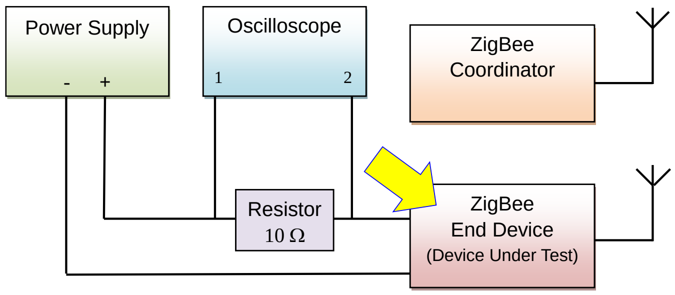


Figure 4. Measurement configuration.

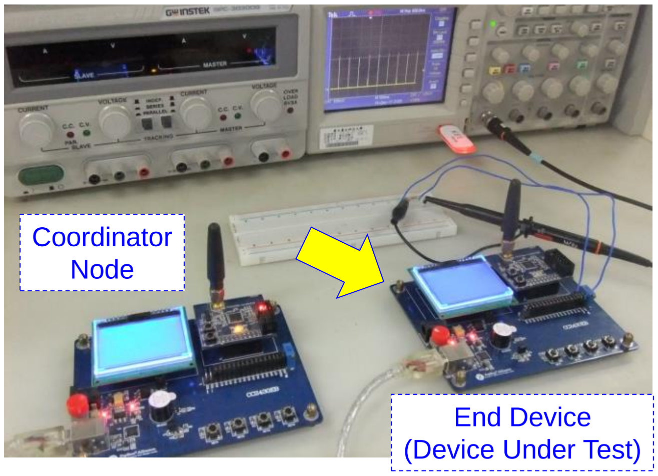
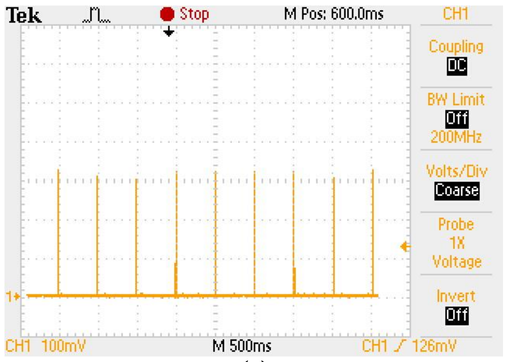


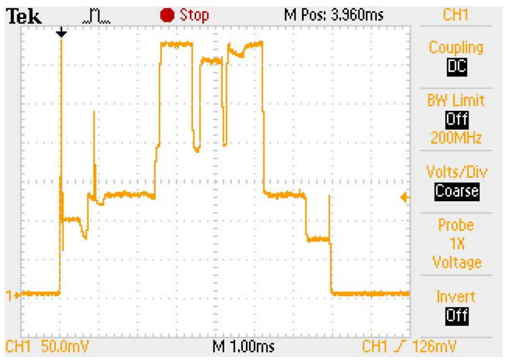
Figure 5. Hardware setup during energy consumption measurement.

B. *Measurement Results*

Figure 6 (a) shows the power consumption during sleep and awake states. The time base on the oscilloscope is set to 500 ms per division, and it can be seen that it is about 0.5 sec among each current peak, showing the power consumption when the device is awake to send the data to the coordinator. Figure 6 (b) is a zoomed version of Figure 6 (a) and shows the current consumption during the active modes in more details. This snapshot has a time base of 1 ms per division. The duration of the active mode is about 7 ms. According to the measurement results, the consumed energy and duration of each operation can be estimated.



(a)



(b)

Figure 6. Measurement results for the division scale at (a) 500 ms and (b) 1 ms.

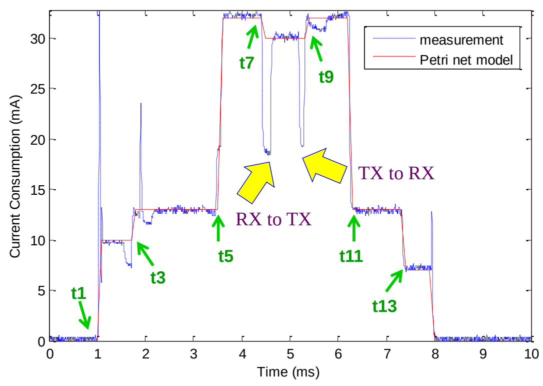


Figure 7. Comparison of energy consumption between measurement and Petri net model.

C. Discussions

With the measured sets of consumed current and duration for each transition as the inputs to the Petri net model, the energy consumption can be obtained as shown in Figure 7. In general, the energy consumption of the Petri net model is close to the practical measurement with a mean difference of less than 1%. However, several peak currents appear during the state transitions, especially the startup sequence t1. The current peaks show the energy consumption when the sensor node is triggered for data transmission. Moreover, note that between transitions t7 and t9, there are two V-shaped gullies, which present the energy consumption of the transceiver turnaround operations, which are the RX to TX and the TX to RX, respectively. Future work would attempt to model such detailed behaviors.

Obviously, the description of the power consumption of a single node during a standard RX/TX procedure is a very isolated scenario. For example, the power consumption of a node would significantly change when a collision is happening during transmission with a subsequent packet loss, which requires a repeated transmission. Further work would consider more practical interactions between the nodes so as to simulate the power consumption of a whole sensor network for different scenarios.

IV. More Experiments Regarding Power Consumption of Wireless Sensor Networks

This section will show more experiments regarding the measurement of power consumption of wireless sensor networks. In this case, the main target is to analyze the different power profiles of a modular WSN hardware platform, which have been designed to reach ultra-low-power states.

A. Developed Cookie Platforms

The implementation of a flexible and configurable processing layer is then compared to an already existing processing solution to show the main benefits of creating a more complex power management profile within the sensor node. The baseline structure of the proposed setup lies on the Cookie platform [15]-[16], which is a modular and fully adaptable hardware architecture mainly composed of four layers: The processing layer, which is the core of the node and includes the microcontrollers of processors to carry out the computation tasks in accordance with the application requirements; the communication layer, which integrates the wireless module to establish an energy-efficient connection with the rest of the participant devices and the base station in the WSN; the sensor layer, which provides the interface with the target environment by including the sensing capabilities to monitor the target physical magnitudes (if needed); and finally the power supply layer, which is in charge of providing the rest of the layers with the required voltage level based on their intrinsic design and operational stages, also serving as an intelligent power manager for more advanced Cookie configurations.

As shown in Figure 8, the vertical connectors that support the modularity of the Cookie allow a very flexible and adaptable prototyping hardware ecosystem to speed up the development and integration of heterogeneous technologies within the same platform. Thus, it fosters the reusability of hardware components with the inclusion of novel techniques for WSN applications.

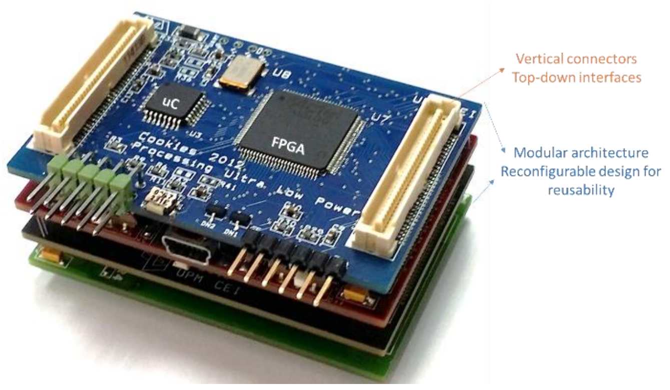

Vertical connectors
Top-down interfaces
Modular architecture
Reconfigurable design for reusability
uC
FPGA
Cookies Processing 2012 Ultra Low Power

Figure 8. Cookie WSN platform composed of 4 modular layers, including an ultra-low-power processing design for very-long-term networks lifetime.

In line with the modular hardware architecture, the Cookie nodes also provide a software support platform to abstract the low-level control of the hardware elements that integrates the sensor devices, as well as speeding up the prototyping of WSN applications and network deployment [20]. The baseline structure of the software layer is composed of a complete set of libraries and functional components that follows the modularity of the hardware layers, including controllers for every processing, sensing, and communication technologies integrated into the Cookie node. These support libraries have been ported to the new design of the ultra-low power processing layer in order to produce a seamless integration of the available hardware elements with the WSN application profiles of the Cookie platform.

The design of this Cookie layer considers three main aspects to exploit the modularity, reconfigurability, and adaptability to different application requirements. The first one relies on the ability of the embedded system to system to adapt to different experimental configurations by allowing for several connectivity arrays of the processing elements to the rest of the platform. This means that, depending on the system requirements, the processing layer can be modified to offer different functional properties. Figure 9 shows the main interconnection structure that allows the definition of several Cookie processing configurations within the same hardware layer.

The second aspect is related to the inclusion of an 8051-based processing architecture, which is in line with the defined HW-SW framework of the Cookies for resource-constrained sensor network applications, and for the efficient implementation of lightweight reprogramming strategies for extending the overall network lifetime [21]. The third aspect lies integrating ultra-low-power hardware elements within the system architecture, seeking the balance between flexibility and power-awareness.

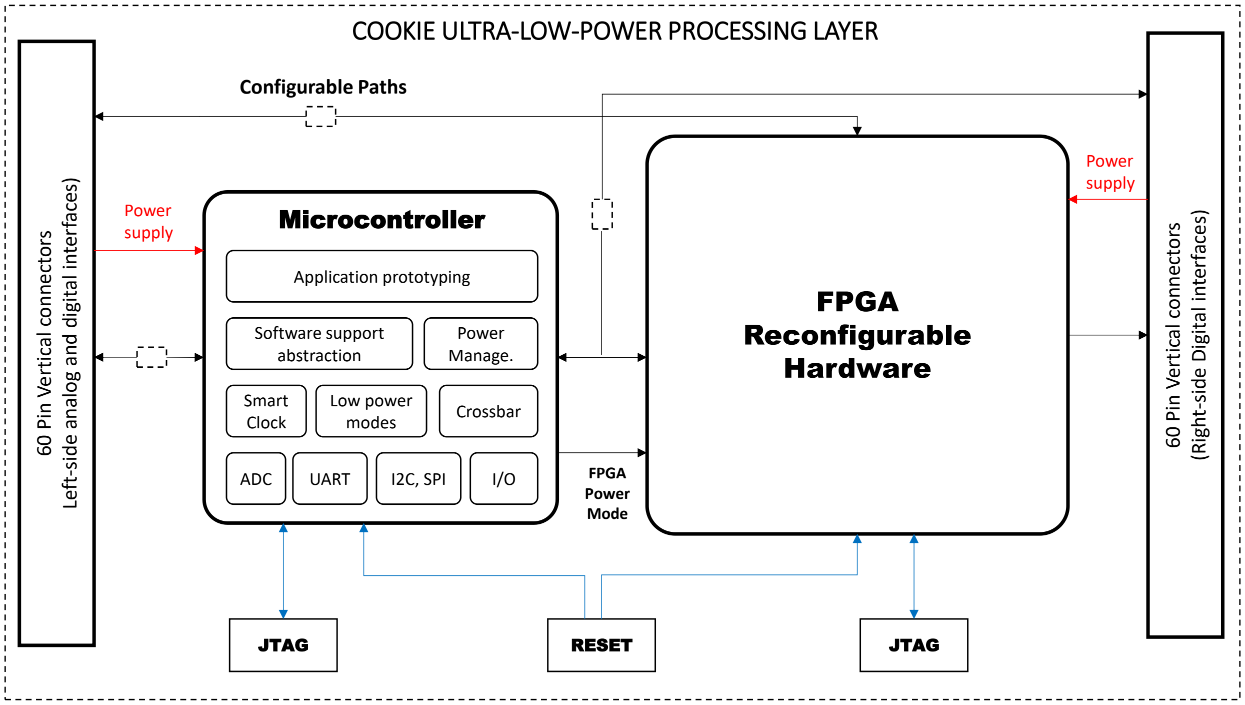

COOKIE ULTRA-LOW-POWER PROCESSING LAYER
Configurable Paths
60 Pin Vertical connectors
Left-side analog and digital interfaces)
Power supply
Microcontroller
Application prototyping
Software support abstraction
Power Manage.
Smart Clock
Low power modes
Crossbar
ADC
UART
I2C, SPI
I/O
FPGA Power Mode
FPGA Reconfigurable Hardware
Power supply
60 Pin Vertical connectors
(Right-side Digital interfaces)
JTAG
RESET
JTAG

Figure 9. Configurable structure of the ultra-low-power Cookie processing layer considering flexibility and adaptability to different resource-constrained application contexts.

Based on the interconnection structure shown in Figure 9, the designed processing layer provides five different operational modes, whose configurations can be made by using bridge connections among layer paths according to the WSN based system constraints definition for the target application/experiment, as follows.

- *Microcontroller with ADC signal connections*: this mode only includes the main processing element and it is thought to be used in those applications where the power consumption is the most critical aspect to be considered. This mode dedicates some of the port lines of the microcontroller for analog input signal processing from the left side of the Cookie interface (for instance, considering the analog sensor from the sensing layer).
- *Microcontroller without ADC signal connections*: this is a variation of the former mode intended to provide the maximum number of digital input/output ports to the rest of the platform.
- *FPGA as the core processor:* in this case, a trade-off between fast signal processing, digital connections availability and power consumption is sought. Thus, the I/O pins of the FPGA are spread between both vertical interfaces of the Cookie platform.
- *Microcontroller with ADC signal connections and FPGA*: this mode offers the widest possibility for platform experimentation in a single layer configuration array, so as to be able to combine the functional component capabilities and peripherals of the microcontroller with hardware block implementations for co-processing and debugging tasks in the FPGA. In this case, both analog and digital sensor signals are contemplated.
- *Microcontroller and FPGA*: It offers a similar approach as the aforementioned mode though more ports of the microcontrollers are dedicated to digital interfaces of the Cookie node.

B. Measurement Setup

As mention before, the experimental analysis is firstly focused on a very configurable processing layer that provides an extended set of different power modes.

Figure 10 shows the experimental setup to evaluate the power consumption of the processing layer based on the configuration of the different power modes depicted in Table II. Thus, several functional scenarios have been set up in order to characterize different power consumption profiles of the designed low-power processing layer, considering not only the various configurations that the hardware layer can adopt, but also the operational modes of the microcontroller and the FPGA elements. To do so, the experimental setup was composed of the implemented Cookie layer plus an extension board that allows supplying the different voltage levels of the modular platform from an external source (as shown in Figure 9), so that no additional consumption offsets can significantly shift the encountered measurements.

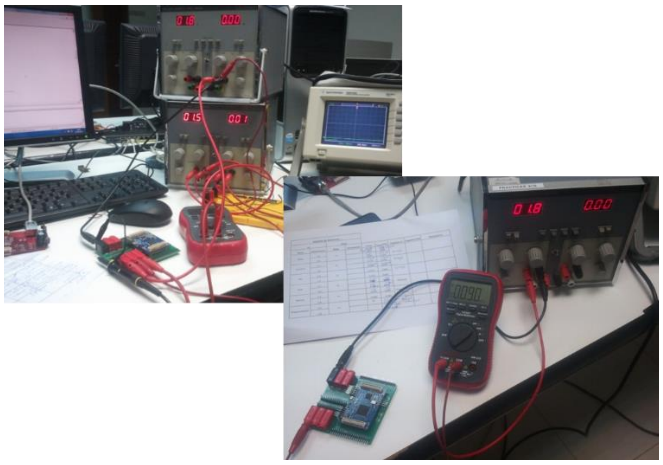


Figure 10. Experimental setup for the characterization of the WSN ultra-low sensor node design.

On one hand, the C8051F930, which is an 8051-based 8-bit microcontroller from Silicon Labs with Crossbar technology [22] is the main processing element that includes the software support platform for managing the Cookie sensor nodes. The C8051F930 is composed of a 64 kB flash program memory, internal and external RAM memories of 256 bytes and 4 kB respectively, 10-bits ADC, four 16-bits timers and Smart Real Time Clock, UART and 2 SPI, and most particularly it supports a voltage supply range of 1.8 V to 3.6 V.

On the other hand, a flash-based FPGA Igloo AGL030V5 from Actel (Microsemi) [23] that serves as a co-processing element for performing faster control tasks that need to be implemented in hardware. The advantage of this technology is the ability to power-down the whole FPGA without losing the hardware implementation (so no need to download the bitstream once the system is power-up again), thus allowing deep power consumption modes without penalizing performance. The AGL030V5 is a 1.2 V to 1.5 V low power flash technology programmable logic device composed of 30000 system gates, 768 D-flip-flops, 81 user I/O pins, and Flash*Freeze sleep mode control without needing to disconnect supply voltages.

The combination of both technologies allows promoting a wider spectrum of configuration, stand-by and functional scenarios that will ultimately have a positive impact in the node lifetime, hence extending the overall long-term operability of the network.

TABLE II
POWER CONSUMPTION RESULTS COMPARING THE DIFFERENT uC POWER MODES AGAINST NORMAL OPERATION, CONSIDERING DIFFERENT CONFIGURATIONS FOR THE MICROCONTROLLER AND THE INPUT VOLTAGE SUPPLY LEVEL

uC		Current consumption		uC configuration	
Operational mode	**Voltage Supply (V2)**	**I_{DOWN} (mA)**	**I_{NORMAL} (mA)**	**uC Code**	**Details**
Sleep	2.5	0.011	2.180	Sleep mode trigger (via smartclock) 846 bytes	Low-power oscilator (20 MHz) / 4 ==>> 5 MHz. Timer 2. SmaRTClock in 40 KHz self-oscillate Mode. Port Match. Comparator 0. II/O port decoder.
	1.8	0.005	2.140		
Suspend	2.5	0.105	2.174	Suspend mode trigger (via smartclock) 847 bytes	Low-power oscilator (20 MHz) / 4 ==>> 5 MHz. Timer 2. SmaRTClock in 40 KHz self-oscillate Mode. Port Match. Comparator 0. I/O port decoder.
	1.8	0.087	1.930		
Idle	2.5	0.520	0.850	Iddle mode trigger (via Timer interruption) 237 bytes	Low-power oscilator (20 MHz) / 8 ==>> 2,5 MHz. Timer 0. I/O port decoder.
	1.8	0.510	0.820		
Stop	2.5	0.500	0.830	Stop mode trigger (via external reset) 208 bytes	Low-power oscilator (20 MHz) / 8 ==>> 2,5 MHz. Timer 0. I/O port decoder.
	1.8	0.500	0.810		
Normal	2.5	1.230	1.230	uC basic test 48 bytes	Low-power oscilator (20 MHz) / 8 ==>> 2,5 MHz. I/O port decoder.
	1.8	0.900	0.900		
Programming	2.5	---	2.74 (Erasing) 4.6 (Writing)	uC basic test 48 bytes	Low-power oscilator (20 MHz) / 8 ==>> 2,5 MHz. I/O port decoder.
	1.8	---	2.58 (Erasing) 4.5 (Writing)		

TABLE III
EXPERIMENTAL RESULTS CONSIDERING THE COMBINATION OF THE SLEEP POWER MODES OF THE FPGA AND THE MICROCONTROLLER

uC		FPGA		I Consumpt. (mA)
Mode	**V2 (V)**	**Mode**	**V1->VCC (V)**	
nc		Normal	1.5	0.094
Normal	1.8	Flash*Freeze	1.5	0.0044 (V1), 0.92 (V2)
Normal	1.8	Normal	1.5	0.154 (V1), 1.05 (V2)

C. Measurement Results

Both Tables II and III summarize the power-mode profile of the ultra-low power consumption layer, where a combination of two main cores produces a fine-grained configuration set. Tests were also performed considering, on one hand, the inclusion of both components into the hardware layer (uC + FPGA) and on the other hand, the FPGA as the only processing element of the board.

Table II shows the results regarding the configuration that only includes the C8051F930, so V2 was used to characterize the current consumption profile of the layer in such conditions, taking into account 1.8 and 2.5V as voltage supply levels. The current consumption in active mode depends on the number of activated peripherals of the microcontroller, whereas the power-down transition modes are triggered by using various sources of activations, as detailed in the uC configuration column. For instance, differences in terms of power consumption in active modes can be distinguished with respect to the ADuC841-based layer, where I_{normal} raises up to 11mA in normal operation (with VDD=2.8V, 11.0592MHz), while 20uA in power down mode.

Table III depicts the current consumption considering V1 = 1.5V for VCC and VCCBIx (3.67MHz clock), V2 = 1.8V for the uC, and the flash*freeze activation for power-down mode on the AGL030v5, whose results clearly show the difference in consumption when switching the FPGA to the ultra-low power mode and the normal operation (around 100uA and 5uA, respectively) both in FPGA and uC+FPGA layer configuration.

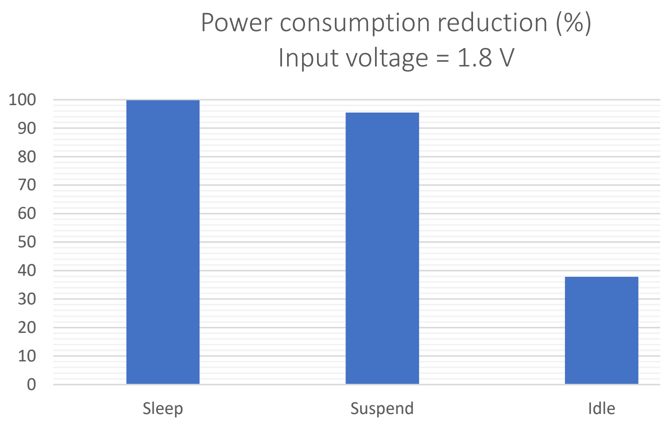


Figure 11. Comparison of the different consumption improvement obtained experimentally for every power mode, and for an input supply voltage of 1.8 V.

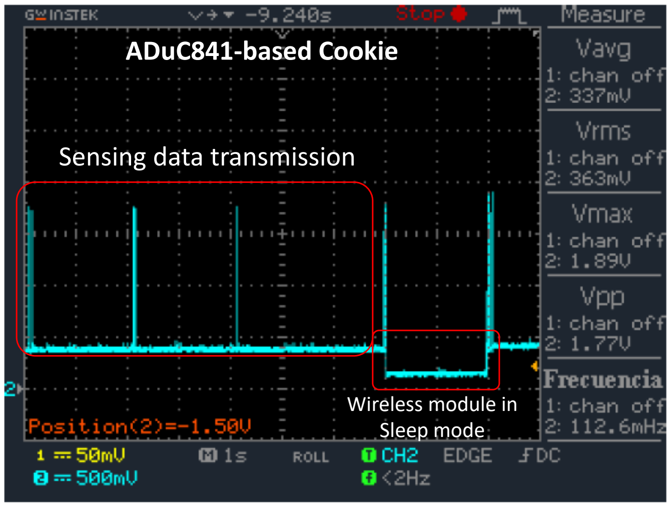
Figure 11 depicts the comparative results regarding the percentage of real consumption of the processing element when considering an input supply voltage of 1.8 V and the three main low-power modes: Sleep, Suspend and Idle. The experimental results show that the sleep modes provides a 99.7% of reduction in comparison with the normal operation (and 94.3% with respect to the next configurable mode), which represents an important improvement for very-long-term operability of the WSN, where the sensor node can stay in ultra-low power mode for longer periods of times. On the other hand, 95.4% and 37.8% improvement are respectively obtained for Suspend and Idle modes, so a trade-off between sensor node activity and energy balance are also possible with these intermediate states.

The comparison of both processing layers for resource-constrained applications has experimentally carried out by measuring the power consumption when performing the transition between wireless data transmission and putting the communication module in sleep mode. Figure 12 shows the results comparing the power consumption in both processing layers, and including the same application code into the microcontrollers (wake-up the wireless communication module, periodic sensing data transmission and putting the module back to sleep mode). The experimental outcomes clearly exhibit the improvement of the ultra-low power design, having a current consumption of 3.23 mA when the wireless module is in sleep mode, whereas the other processing layer obtains a value of 24.9 mA.

D. Discussions

The combination of the intrinsic flexibility of the modular hardware platform with the design and implementation of an ultra-low power processing layer provides important benefits in terms of energy savings as well as a trade-off between computing capabilities and long-term lifetime awareness. The reduction of the power consumption in every mode is very noticeable when comparing first with the next configuration step (obtaining more than 90% in the deepest low-power modes, that is, suspend and sleep, and more than 80% between idle and suspend) and second with a more general-purpose (microcontroller + FPGA) design without particular ultra-low power strategies (up to 87 % depending on the configuration to be adopted). The results show that it is indeed possible to refine the power-computing balance both at hardware and software levels without a strong penalization in one of the figures of merits sought. For instance, a very extreme configuration based on the only microcontroller and running with 1.8 V can be applicable in some use cases, although shifting to another configuration point may be possible if considering the experimental outcomes against the desired power consumption boundaries. Therefore, a more computational-effective solution without penalizing the energy cost is feasible.

In this sense, the level of reconfigurability and adaptability can be highlighted in three ways, according to the application needs: Selection of the processing elements to be included and their interfaces in line with the five supported combinations; the power supply levels in accordance with the threshold limits and considering the results of the power consumption experiments for every operational state; and the management of the different low-power modes depending on the duty-cycle and QoS requirements for the target scenario. While for the first two cases the decisions can be taken at design and pre-deployment time (although dynamic voltage scaling might be a possibility to be considered as well), the latter exhibits a very powerful opportunity to reduce power at runtime and in an adaptive fashion.



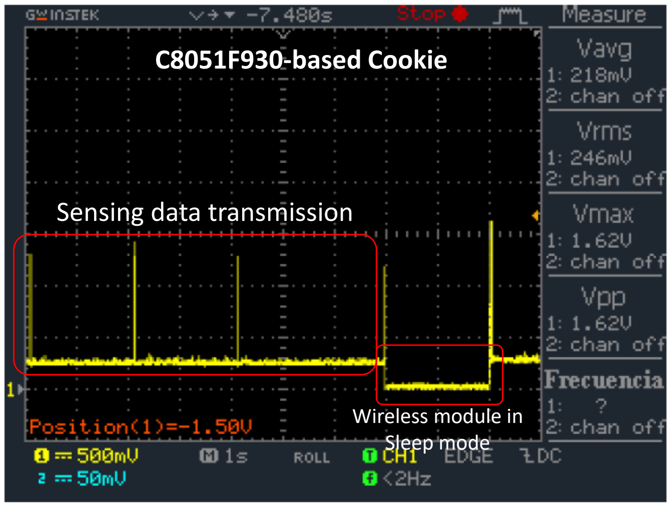


Figure 12. Experimental comparison of two processing layers when performing the same WSN application and power modes transitions.

V. Conclusion and Future Work

In this paper, a systematic approach to modeling and measurement of energy consumption for wireless sensors has been presented. In the prior work, the sensor operation is modeled using the Petri nets. Then, a preliminary experiment has been conducted to measure the real power consumption and provide input parameters to the Petri net model. The comparative results indicate the Petri net model has approximated the real measurement under the assumed scenarios. Besides the periodical operations demonstrated in this paper, the measurement scheme is also useful for other specific applications and could be fed back to the Petri net model as a calibration source. Since the proposed Petri net model in this paper is mainly designed for packet transmission and reception, as future work, operations of sensing and computation tasks could be further considered so as to make the model much more realistic. Also, with a given battery, the proposed model could be further applied to the lifetime estimation for periodical operations.

Furthermore, more experiments regarding measurements of power consumption for wireless sensor networks have conducted to analyze different power profiles to reach ultra-low-power states by using the developed modular Cookie platforms. Also, one direction of future work is the modeling and measurement of a variety of sensors for power consumption under the structure of the Internet of things [24]. It is believed that the experimental measurements presented in this paper would benefit application engineers in analyzing and understanding the power consumption of wireless sensor networks.

Acknowledgment

This paper was supported by the Ministry of Science and Technology (MOST) of Taiwan under grant MOST-106-2221-E-027-057. The authors and specifically Jorge Portilla would like to thank the "José Castillejo" program for junior postdoc research stays, funded by the Ministry of Education, Culture and Sports of the Government of Spain, grant number CAS17/00259. Also, the authors would like to thank Mr. Yu-Kai Wang from Information and Communications Research Labs, Industrial Technology Research Institute (ITRI), Taiwan, for his help with part of the experiments.

References

[1] J. S. Lee and Y. H. Sun, "Behavior modeling of networked wireless sensors for energy consumption using Petri nets," in *Proc. IARIA International Conference on Systems and Networks Communications (ICSNC)*, Nice, France, October 2018, pp. 64-68.

[2] J. S. Lee and Y. C. Lee, "An application of grey prediction to transmission power control in mobile sensor networks," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2154-2162, Jun. 2018.

[3] W. Q. Guo, W. M. Healy, and M. C. Zhou, "Impacts of 2.4-GHz ISM band interference on IEEE 802.15.4 wireless sensor network reliability in buildings," *IEEE Trans. Instrum. Meas.*, vol. 61, no. 9, pp. 2533-2544, Sep. 2012.

[4] J. S. Lee and W. L. Cheng, "A fuzzy-logic-based clustering approach for wireless sensor networks using energy predication," *IEEE Sensors J.*, vol. 12, no. 9, pp. 2891-2897, Sep. 2012.

[5] J. S. Lee, "A Petri net design of command filters for semi-autonomous mobile sensor networks," *IEEE Trans. Ind. Electron.*, vol. 55, no. 4, pp. 1835-1841, Apr. 2008.

[6] G. Anastasi, M. Conti, M. D. Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Netw.*, vol. 7, no. 3, pp. 537-568, May 2009.

[7] J. S. Lee and T. Y. Kao, "An improved three-layer low-energy adaptive clustering hierarchy for wireless sensor networks," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 951-958, Dec. 2016.

[8] J. S. Lee and C. L. Teng, "An enhanced hierarchical clustering approach for mobile sensor networks using fuzzy inference systems," *IEEE Internet Things J.*, vol. 4, no. 4, pp. 1095-1103, Aug. 2017.

[9] B. Bougard, F. Catthoor, D. C. Daly, A. Chandrakasan, and W. Dehaene, "Energy efficiency of the IEEE 802.15.4 standard in dense wireless microsensor networks: Modeling and improvement perspectives," in *Proc. Design, Automation and Test in Europe Conf. and Exhibition (DATE)*, Munich, Germany, Mar. 2005, pp. 196-201.

[10] M. C. Ekstrom, M. Bergblomma, M. Linden, M. Bjorkman, and M. Ekstrom, "A Bluetooth radio energy consumption model for low-duty-cycle applications," *IEEE Trans. Instrum. Meas.*, vol. 61, no. 3, pp. 609-617, Mar. 2012.

[11] J. R. B. Sousa, A. M. N. Lima, and A. Perkusich, "Modeling and analyzing power consumption in sensor networks nodes based on differential hybrid Petri nets," in *Proc. IEEE Conf. Industrial Electronics Society (IECON)*, Raleigh, NC, Nov. 2005, pp. 389-394.

[12] P. S. Sausen, J. R. B. Sousa, M. A. Spohn, A. Perkusich, and A. M. N. Lima, "Dynamic power management with scheduled switching modes," *Comput. Commun.*, vol. 31, pp. 3625-3637, Sep. 2008.

[13] A. Shareef and Y. F. Zhu, "Power modeling of wireless sensor nodes based on Petri net," in *Proc. The 39th Int. Conf. Parallel Processing*, San Diego, CA, Sep. 2010, pp. 101-110.

[14] J. S. Lee, M. C. Zhou, and P. L. Hsu, "Multiparadigm modeling for hybrid dynamic systems using a Petri net framework," *IEEE Trans. Syst. Man Cybern. A., Syst. Humans*, vol. 38, no. 2, pp. 493-498, Mar. 2008.

[15] J. Portilla, T. Riesgo, A. Abril, and A. de Castro, "Rapid prototyping for multi-application sensor networking", *SPIE Newsroom*, Nov. 2007, DOI:10.1117/2.1200711.0851.

[16] G. Mujica, R. Rodriguez-Zurrunero, M. R. Wilby, J. Portilla, A. B. Rodríguez González, A. Araujo, T. Riesgo, and J. J. Vinagre Díaz, "Edge and fog computing platform for data fusion of complex heterogeneous sensors," *Sensors*, vol. 18, no. 11, article ID 3630, 2018.

[17] M. C. Zhou, K. Venkatesh, *Modeling Simulation and Control of Flexible Manufacturing Systems: A Petri Net Approach*, Singapore:World Scientific, 1998.

[18] J. S. Lee and P. L. Hsu, "Implementation of a remote hierarchical supervision system using Petri nets and agent technology," *IEEE Trans. Syst. Man Cybern. C. Appl. Rev.*, vol. 37, no. 1, pp. 77-85, Jan. 2007.

[19] B. Selvig, “Measuring power consumption with CC2430 and Z-Stack,” *Application Note AN053*, Texas Instruments. Jul. 2007.

[20] G. Mujica, V. Rosello, J. Portilla, and T. Riesgo, “Hardware-software integration platform for a WSN testbed based on cookies nodes”, in *Proc. IEEE Conf. Industrial Electronics Society (IECON)*, Montreal, Canada, Oct. 2012, pp. 6013-6018.

[21] G. Mujica, V. Rosello, J. Portilla, and T. Riesgo, “On-the-fly dynamic reprogramming mechanism for increasing the energy efficiency and supporting multi-experimental capabilities in WSNs,” in *Proc. IEEE Conf. Industrial Electronics Society (IECON)*, Vienna, Austria, Nov. 2013, pp. 5455-5460.

[22] Silicon Labs, *C8051F92x-F93x Ultra-Low-Power MCUs*. [Online] Available: http://www.silabs.com. [Accessed: 10-May-2019].

[23] Microsemi, *IGLOO Low-Power Flash FPGAs Family,* [Online] Available: http://www.microsemi.com/products/fpga-soc/fpga /igloo-overview. [Accessed: 10-May-2019].

[24] J. Portilla, G. Mujica, J. S. Lee, and T. Riesgo, “The extreme edge at the bottom of the Internet of things: A review,” *IEEE Sensors J.*, vol. 19, no. 9, pp. 3179-3190, May 2019.

Optimal Control of Multicasting Multimedia Streams in Cloud Data Centers Networks

Nader F. Mir, Abhishek Rao, Rohit Garg, Mitesh Ambavale, and Raviteja Ainampudi
Department of Electrical Engineering
San Jose State University, California
San Jose, CA, 95195, U.S.A
email: nader.mir@sjsu.edu

Abstract - **The objective of this paper is a thorough study on the optimal control of multicasting multimedia streaming over the components of data center networks. Such a network control includes finding the best locations of traffic multicasting, assigning the quality of service (QoS) while multicasting, and the right network devices for this objective. Since streaming media accounts for a large portion of the traffic in networks, and its delivery requires continuous service, it is essential to analyze the traffic sources and investigate the network performance. IP multicasting technology adds tremendous amount of challenge to a network as the streamed media delivered to users must be multiplied in volume. In this paper, we first demonstrate a study of the complexity and feasibility of multicast multimedia networks by using different multicast protocols and video sources. In our proposed method, we then create peer-to-peer (P2P) and data center topologies in order to analyze the performance metrics. The implementation and evaluation of the presented methodology are carried out using OPNET Modeler simulator and the various built-in models. Further, we implement performance tests to compare the efficiency of the presented topologies at various levels. At the end of the paper we analyze the optimal locations for multicasting multimedia streaming traffic.**

Keywords - video streaming; cloud data centers; multicast; multimedia; performance evaluation; video codecs; IP; MPLS.

I. INTRODUCTION AND BACKGROUND

In [1], an efficient video-based packet multicast method for multimedia control in cloud data center networks was presented. The current paper extends that work and conducts a thorough study toward several other aspects, such as Comparison of throughput and delay at switching nodes on the control of multicasting multimedia streaming over the components of data center networks.

Over the years, major development in the industry have been involved in the integration of various multimedia applications. Delivery of streaming media (video on demand), e-learning with minimum delay and highest quality has been one of the major challenges in the networking industry. Video service providers, such as Netflix, Hulu are constantly changing the architecture in order to service these needs. These service providers face stiff competition and pressure to deliver the next generation of streaming media to the subscribers. The next generation media can be divided into categories: real-time and non-real time. Examples of real time can be live streaming and video conferencing and non-real time can be e-learning and video on demand [2]. The next generation of streaming media [3] involves a large number of subscribers whose delivery is closer aligned with the latest protocols than with the traditional systems. In such cases, it is required that the service providers upgrade their infrastructure and support them [4].

One of the main challenges in the multimedia industry that motivates us to look into it in this paper is multicasting the video streams. IP Multicast is one of the major techniques that can be used for efficient delivery of streaming multimedia traffic to a large number of subscribers simultaneously. Group membership, unicast and multicast routing protocols are mainly required for multicast communications [5]. *Inter Group Membership Protocol* (IGMP) utilized in our study maintains one of the most commonly used multicast protocols at user facility site. IGMP is used to obtain the multicast information in a network. Unicast routing protocols can be either *distance vector* or *link state*; the latter being preferred due to the dynamic reaction of these protocols to changes in topology. Multicast routing protocols can be integrated with the unicast routing protocol or can be independent of them. Protocols, such as the *Multicast Open Shortest Path First* (MOSPF), depend on the underlying unicast protocols used, whereas protocols such as *Protocol Independent Multicast* (PIM), are independent of the type of unicast routing protocols used. A combination of IGMP, MOSPF and PIM in sparse mode or dense mode can be used for successful implementation and efficient delivery of multimedia traffic in networks [6].

Multicast routing enables transmission of data to multiple sources simultaneously. The underlying algorithm involves finding a tree of links connecting to all the routers that contain hosts belonging to a particular multicast group. Multicast packets are then transmitted along the tree path from the source to a single destination or a group of receivers belonging to a multicast group. In order to achieve the multicast routing tree, several approaches have been adopted. Group-shared tree, source based tree and core based tree are some which are explained here.

a. *Group-based tree*: In this approach a single routing tree is constructed for all the members in the multicast group;
b. *Source-based tree*: This involves constructing a separate routing tree for each separate member in the multicast group. If multicast routing is carried out using source-

based approach, then N separate routing trees are built for each of the N hosts in the group [7]; and

c. *Core-based tree*: This is a multicast routing protocol, which builds the routing table using a group-shared tree approach. The tree is built between edge and core routers in a network, which helps in transmitting the multicast packets.

MOSPF and PIM use one of the above mentioned approaches in the transmission of packets. As PIM is the multicast routing protocol used in the implementation, we discuss the working of PIM.

PIM is a multicast routing protocol that is independent of the underlying unicast routing protocols used [8]. PIM works in two modes dense mode and sparse mode. In the former mode the multicast group members are located in a dense manner and the latter approach has the multicast group members distributed widely. PIM uses Reverse path forwarding (RPF) technique in dense modes to route the multicast packets. In dense mode, RPF floods packets to all multicast routers that belong to a multicast group whereas in a sparse mode PIM uses a center based method to construct the multicast routing table. PIM routers which work in sparse mode send messages to a center router called rendezvous point. The router chosen to be rendezvous point transmits the packets using the group based tree model. As seen in Fig. 1, the rendezvous point (RP) can move from a group-based tree model to a source-based approach if multiple sources are specified [9][10].

A broadband network is a communication infrastructure that can provide higher bandwidth services. Regional networks are connected to such broadband backbone networks to form the Internet. Internet Service Providers own the regional broadband networks. A broadband network is required to support the exchange of multiple types of information such as, massive data storage access, voice over IP (VoIP), video streaming, and live multicasting, while satisfying the performance requirements of each application. In short, the delay and jitter should be minimum for better performance of these applications.

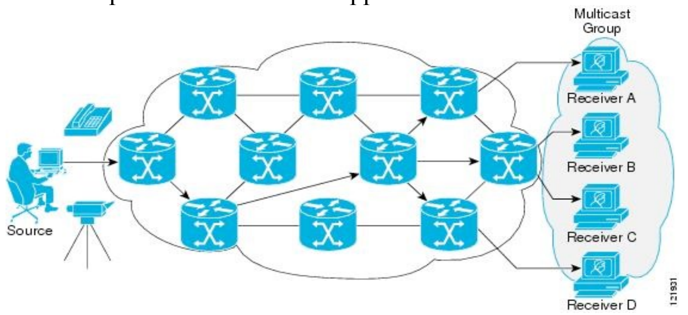


Figure 1. Sample diagram of multicast routing

The above-stated requirements are satisfied by employing broadband routers and switches, fiber optic cables, and tunneling mechanisms. Tunneling makes the communication faster in broadband networks compared to normal routing mechanism. *Multiprotocol label switching* (MPLS) is networking infrastructures used in a high-speed backbone network to provide a better quality of experience for broadband applications such as, video streaming and other real-time applications.

In any MPLS network, the routers at the edge of the network are the most complex ones. In edge routers, user services such as policies, rate limiters, logical circuits, and address assignment are created. In a certain connection between a pair of users, edge routers keep a clean separation between a complex edge and create services while routers in the middle mainly do basic packet forwarding by switching MPLS packets from one interface to the other. The MPLS header (also known as the label) is imposed between the data link layer (layer 2) header and network layer (layer 3) header.

MPLS adds some traditional layer 2 capabilities and services, such as traffic engineering, to the IP layer. The separation of the MPLS control and forwarding components has led to multilayer, multiprotocol interoperability between layer 2 and layer 3 protocols. MPLS uses a small label or stack of labels appended to packets and typically makes efficient routing decisions. Another benefit is flexibility in merging IP-based networks with fast-switching capabilities.

In MPLS networks, any IP packet entering the MPLS network is encapsulated by a simple header called a *label*. The entire routing is therefore based on the assignment of labels to packets. The compelling point of MPLS is that this simple label is always processed for routing instead of the IP header in the network. Note that labels have only local significance. This fact removes a considerable amount of the network-management burden.

The rest of this paper is organized as follows: Section II provides a detail of our architecture and its functionality and Section III presents a performance analysis of the designed architectures. Finally, Section IV concludes the paper.

II. NETWORK ARCHITECTURE

Network architecture has been designed from service provider's and user's perspective. Network service providers are concerned with the available bandwidth and utilization of resources whereas end user's main concern is with the delivery of streaming media with lowest time and maximum efficiency. In order to obtain the various parameters that are required for the best design of multimedia network, two network models were implemented and analyzed.

A. *Implemented Peer to Peer (P2P) Network Design*

Peer to peer network model is a distributed architecture where the application is transmitted between source and destination through peers. Applications such as music sharing, file sharing use peer-to-peer network model for transmitting the data. A peer-to-peer network was built using the values as shown in Table I. The network architecture shown below represents an organizational division where the admin department is the source of multimedia traffic, which is simultaneously streamed to the remaining departments namely the HR, finance and IT. The topology contains two backbone routers connected back-to-back, a video streaming source is configured and stored in the admin department, where the video frames are encoded with a H.264 codec and generating a frame rate of 15-20 frames per sec. The backbone routers are configured with PIM-DM as the multicast protocol that is responsible to carry multicast packets.

TABLE I. CONFIGURATION PARAMETERS FOR A PEER TO PEER NETWORK DESIGN

Link speed (in Mbps)	Frame size	Frame interarrival rate	Video Codec	Multicast protocol used
100	128x120	10 fps	H.264	PIM-DM
100	128x240	15fps	H.264	PIM-DM
1000	352x240	30 fps	H.264	PIM-DM

B. Implemented Data Center Topology

The data center and its network testbed topology [12] implemented in this paper are shown in Fig. 2. A data center contains certain facilities for computing, data storage, and other technology resources, as shown by "server racks." In the network testbed topology, the interconnections among the switches are regular. The figure shows a data center network using four layers of switches as two layers of *core switches,* one layer of *aggregate switches,* and one layer of *edge switches*. An edge switch, also called top-of-rack (ToR) switch, directly connect to end servers at server racks, and core switches directly connect to the routers such as Router1 and Router2 that are attached to the outside of the data center or directly to the Internet. In this figure, two groups of 6 destinations are considered in the testbed: group A destinations and group B destinations.

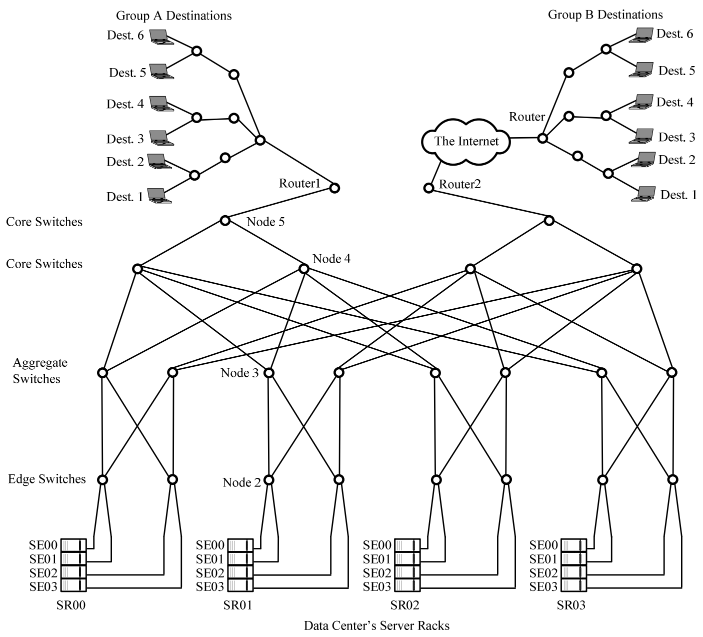


Figure 2. Data center topology as the testbed for multimedia streaming

The topology has been implemented taking into account redundancy at all levels, and responds dynamically to failures at link, path and device level. Scaling the number of nodes, both horizontally and vertically has been considered in order to analyze the performance metrics of the network. Streaming media content stored at the servers are configured for varying bit rates and varying frame sizes.

The OPNET simulator has been used as the simulation tool for implementing and testing multicast multimedia traffic. The detailed metrics that are used for data center implementation has been shown in the Table II.

TABLE II. CONFIGURATION PARAMETERS FOR A DATA CENTER

Number of servers per rack	2
Number of TOR switches used per rack	2
Number of distribution switches per rack	2
Number of core switches per rack	1
Total number of servers	8
Total TOR Switches	8
Total distribution switches	4
Total Core switches	2
Link speeds in data centers	1000 Mbps
Link speeds to WAN	PPP DS3
Video Application and codec used	Video streaming, H.264
Frame sixe	Constant (5000)
Bit rates	Constant (10 fps)

III. PERFORMANCE ANALYSIS

The configuration parameters for used for the performance evaluation are shown below in Table III. Since backbone routers are majorly involved in the transmission of traffic over the internet, Ethernet load across these links has been considered. As the frame size increases load across the backbone links increases, which leads to increase in the delivery of media to destination.

TABLE III. VIDEO CONTENT CONFIGURATION PARAMETERS

Test Name	Frame Size (in bytes)	Video Codec Used	Frame Inter-arrival Time	Ethernet Load Across the Link (packets/sec)
Video1	15,360	H.264	10 Frames/sec	280
Video2	5,000	H.264	Exponential	530

A. Study on Ethernet Load

In order to reduce the end to end delay, latency and prioritized traffic using quality of service (QoS) was implemented. Opnet simulator has various built-in QoS profiles, some of them being WFQ, FIFO, priority queueing. Differentiated services code point based QoS is being used in this implementation wherein based on the priority of traffic delivery, a certain level of service is configured depending on which resources are allocated along the path of delivery.

Now, we present the Ethernet load test – a performance metric which determines the amount of data packets that are carried by the network. Although each link in the network carries data packets WAN / core routers are chosen for analysis. In peer-to-peer topology mentioned earlier, the links connecting the backbone routers are considered, whereas in a data center topology core router links/WAN links have been chosen.

The variation in the graph can be explained as follows. In this case the bit rate and frame size s, both have been kept as exponential increasing functions. From Fig. 3 it can be observed that in a two-node network since there is a single link connecting the backbone routers, Ethernet load across these links is considerably higher than that of a multi node model where, PIM builds a tree structure (source based, or center based) for sending the multicast packets. As a result, the load is distributed across various links thereby reducing the failure percentage. One more alternative that can be used is port-channel can be configured to distribute the load across the links connecting the routers. Over the time considered it was observed that the load was higher in a two-mode network and lesser in a multi node network.

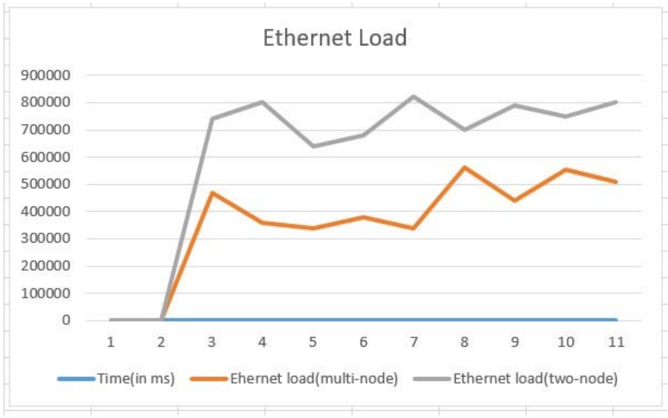


Figure 3. Comparison of Ethernet load between two nodes and multi-mode cases

Our next experiment is concerned with the queueing delay which is the amount of time that a packet waits in the router's queue before being sent onto the network. This is one of the most important parameters for multicast networks as an increase in the queueing delay can cause significant delay in the transmission of packets across the network. Queueing delay can be due to many factors, such as buffer size in a router, router's processing capacity, link speed used, number of hops from source to destination. In this analysis, the queueing delay has been analyzed for a two node and a multi-node environment. From the graphs shown in Fig. 4, it can be observed that although in a two node network links of higher speed were used, when packets of multiple applications arrive, a two-node network experienced significant queueing delay which led to the delay in the transmission of packets. Since no QoS was configured all the packets were serviced based on packet arrival times. The graph for a 2-node network shows peaks of highest queueing delay and lows of least queueing delay. This is due to the fact that when packets related to multiple applications arrive there has been peaks of high queue delay and when packets related to single applications arrive less queueing delay has been experienced. In order to have less queueing delay priority traffic can be classified based on QoS policies which helps in serving these packets better.

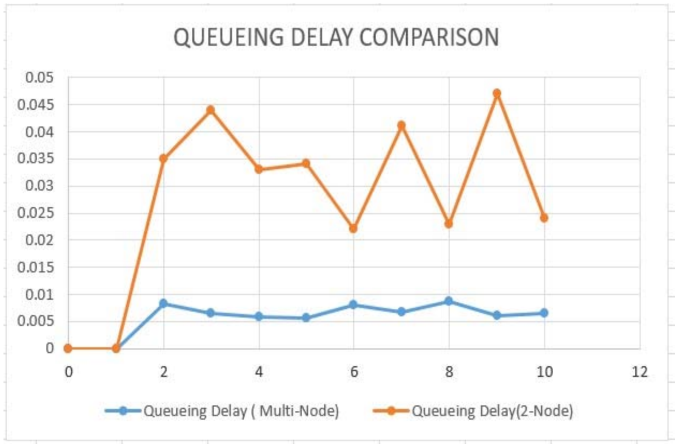


Figure 4. Comparison of queueing delay between two nodes and multi-node cases

Next, we consider a test on QoS, a mechanism which is used to analyze the performance of networks. QoS policies configured ensures traffic prioritization and reservation of resources along the path from source to destination. QoS plays a major role in multimedia networks where defining QoS policies defines the traffic priority when real time multimedia traffic and interactive media is involved. Since these types of traffic have rigid delay constraints defining QoS policies for these types can result in prioritizing them when requests for other traffic are in queue.

Simulation results of QoS implementation is shown in Fig. 5. Since real time interactive media could not be created in a simulation environment, two video sources (video1 and video2) were created and video1was configured with a WFQ QoS profile traffic group of video 1 being set to high priority and traffic group of video 2 being set to best effort with no QoS configured. From the plots in the figure, it can be observed that over a period of time when requests arrive for video1 and video2 packets requesting information, video1 is serviced with less packet delay than those packets for video2 while multicast flow is also included in the configuration. Since the IGMP convergence time was 2 min the QoS traffic servicing has started after the first few minutes.

Finally, the latency is our last performance metric to focus on. The latency is the amount of delay involved in transmitting the data from source to destination. For calculating the Latency issues in network different pixel sizes were chosen for analysis. Three different Pixel sizes were configured over a period of time with link speed and other parameters being kept constant. The link speed was defined to be 100 Mbps and pixel sizes of 352x240, 128X240 and 128X120 were defined with frame interarrival rates to be logarithmic. After several tests it was observed that the latency in the transmission of a high-quality video was more compared to the latencies of the transmission of a video of lesser resolution as shown in Fig. 6. If a video of high quality has to be transmitted in minimum time, then separate channels can be used for high definition video where source specific trees can be used for routing thereby achieving successful routing of packets.

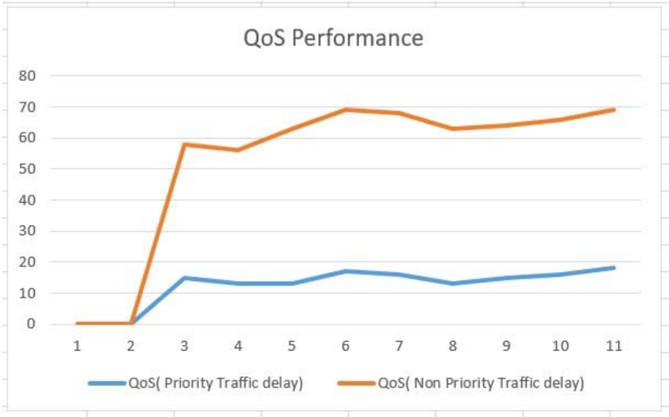


Figure 5. QoS servicing of priority and non-priority traffic

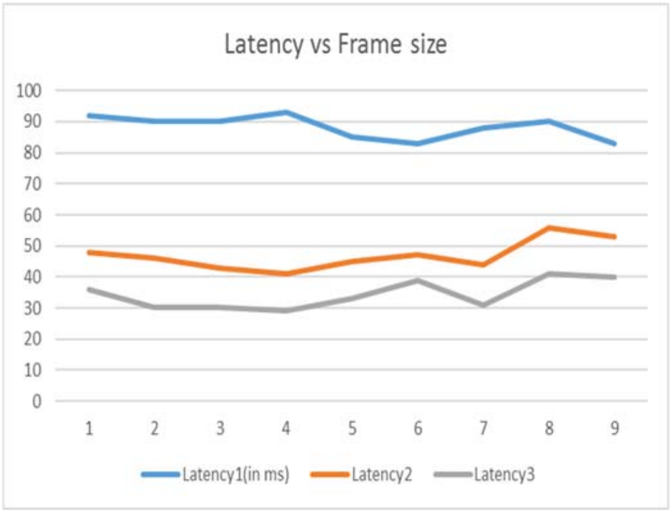


Figure 6. Comparison of latency of various frame sizes.

B. *Study on Network Infrastructure for Media Multicasting*

An MPLS network consists of nodes called *label switch routers* (LSRs). An LSR switches label packets according to its particular *forwarding tables*. The content of a forwarding table consists of labels to be assigned to a flow of traffic. An LSR has two distinct functional components: a control component and a forwarding component. The control component also facilitates the exchange of information with other LSRs to build and maintain the forwarding table.

A typical advanced router or switch has routing tables in its control plane and an IP forwarding table in its data plane for routing purposes. The routing table of such a device receives signaling packets such as RIP or OSPF packets and must update its IP forwarding table occasionally. An LSR has a separate forwarding table to store labels. The MPLS forwarding table interacts with the IP forwarding table in order to arrange the conversion of IP address to labels or vice versa.

Fig. 8 shows a basic comparison of streaming in IP and MPLS networks. In Fig. 7 (a), a typical IP network is shown where a source host, as host 1, connects to a destination host, as host 2. The generated IP packets enter a wide-area IP network at edge router R1 and pass over the network using all the relative routing protocols, and eventually reach edge router R2 from which they exit. In Fig. 7 (b), an MPLS-enabled network is contrasted with the IP network; any IP packet entering the MPLS network is encapsulated by a label. The entire routing is therefore based on the assignment of labels to packets. We will see in the next sections how this technique would substantially reduce the time for packet processing and routing.

Assigning labels to each packet makes a label-swapping scheme perform the routing process efficiently and quickly. In Fig. 7 (b), the edge LSRs of the MPLS network are ingress LSR1 and egress LSR2. It can be seen that a label is indeed a header processed by an LSR to forward packets. The header format depends on the network characteristics. LSRs read only labels and do not engage in the network-layer packet headers. One key to the scalability of MPLS is that labels have only local significance between two devices that communicate. When a packet arrives, the forwarding component uses the label of the packet as an index to search the forwarding table for a match. The forwarding component then directs the packet from the input interface to the output interface through the switching fabric.

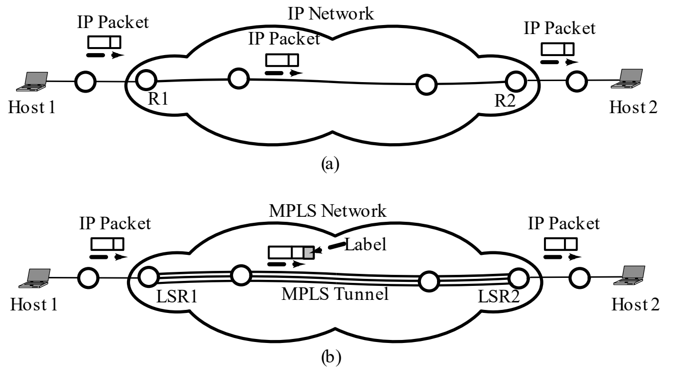


Figure 7. Operations of streaming in MPLS compared with IP infrastructures

The advance mutual agreement between two adjacent LSRs on a certain label to be used for a route is called *label binding*. Once an IP packet enters an MPLS domain, the ingress LSR processes its header information and maps that packet to a *forward equivalence class* (FEC). Any traffic is thus grouped into FECs. An FEC indeed implies that a group of IP packets are forwarded in the same manner—for example, over the same path or with the same forwarding treatment. A packet can be mapped to a particular FEC, based on the following criteria:

- Source and/or destination IP address or IP network addresses
- TCP/UDP port numbers
- Class of service
- Applications

A *label switched path* (LSP) through the network must be defined, and the QoS parameters along that path must be established. An LSP resembles a tunnel. Label usage for identifying the next hop destination instead of the IP destination address adds superior capabilities like traffic engineering to the traditional IP routing. LSRs consist of two functional components; a control component and a forwarding component. The control component uses standard routing protocols to exchange information between the LSRs and facilitates the forwarding table formation. Based on the information in the forwarding table, the forwarding component of the LSR performs the switching of label packets.

OPNET Modeler provides a global development environment capable of infrastructure networks, such as MPLS, modeling and performing discrete event simulations. Data collection and analysis, incorporated along with the design simulation, helps in evaluating and optimizing real life networking topologies. The advantages of OPNET Modeler over other modelers include a simple GUI interface and network modules integrating extensive protocol suite with queuing functionalities.

The 'System in the Loop' module included in this Modeler provides an efficient method to capture live transmission and analyze the behavior of different real-life scenarios. Applications include LAN and WAN performance modeling, network planning and protocol research.

The study on the type of network infrastructure for media streaming involves the performance evaluation of video streaming using MPLS and OPNET Modeler. The OPNET Modeler has the modules to implement the MPLS over the basic network topology. Creating dynamic and static LSPs, Traffic engineering, and Differential Service functionality could be achieved using the Modeler functionalities. The main steps involved in implementing MPLS using the simulator are:

- Configuring "MPLS config" Object
- Establishing MPLS Path Model
- Configuring MPLS parameter on MPLS routers
- Define Neighbor configuration
- Establish the Traffic mapping configuration
- Configuring IP Traffic Demand
- Configure Routing Protocol
- Update MPLS LSP details

Fig. 8 shows a global network perspective for the study on the backbone networking infrastructure used in the simulator. The network topology consists of the following elements: LER1, LSR1, LSR2, LSR3, LSR4, LSR5, LSR6, LER2, and LER2 are Cisco 7200 series routers, which support MPLS protocol along with the standard routing protocols such as, Open Shortest Path First (OSPF) and Routing Information Protocol (RIP).

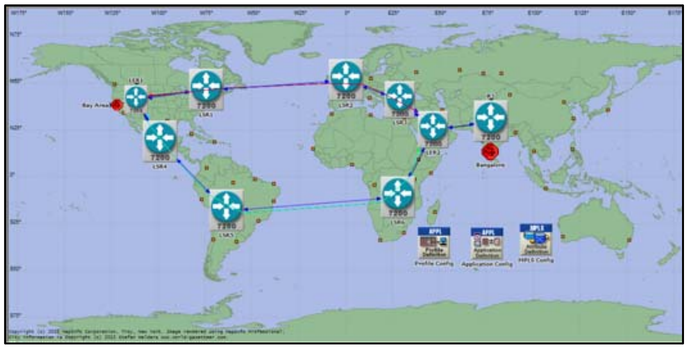


Figure 8. Global network for the study on the backbone networking infrastructure

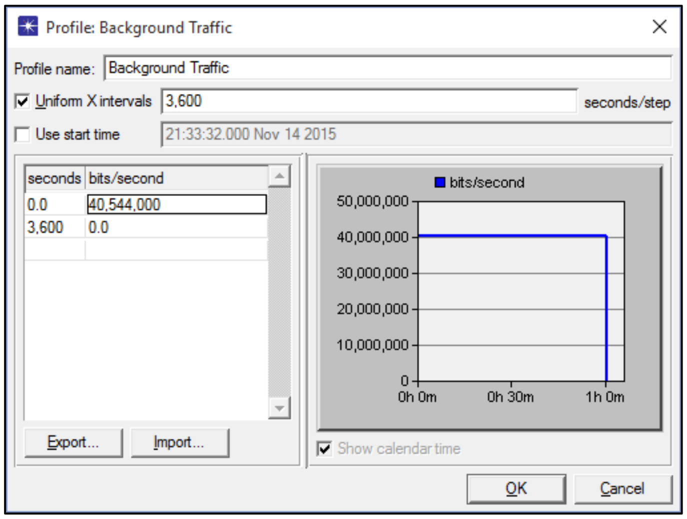
Two subnets were considered in the simulator: San Francisco Bay area subnet and Bangalore subnet. The San Francisco Bay area subnet consists of a Cisco 4000 series router and an Ethernet workstation capable of video conferencing. The Bangalore subnet consists of a Cisco 4000 series router and an Ethernet workstation capable of video conferencing. Unique IPv4 addresses were assigned to each node in the network. All routers in the main subnet are connected via Point-to-Point DS3 links with a data rate of 4.736 Mbps. The router and Ethernet workstation in each subnet are connected using 10Gbps Ethernet links. The simulation objects used in this project are the following:

1. Application Configuration Object
2. Profile Configuration Object
3. MPLS Configuration Object
4. IP Traffic Flow Object

An application configuration object is used in the simulation to define the parameters for the video conferencing application. This object is a video conferencing application so that it receives the highest priority among all the other traffic in the network.

In comparison with the MPLS traffic, we also used an IPv4 traffic flow object that represents the IP layer traffic flow between a specified source and a destination. This object is used to create background traffic in the network. Using this object, a background traffic flow at the approximate rate of 40.5 Mb/s is created. Fig. 9 shows the sample configuration for IP traffic flow. The background traffic flow starts when the simulation starts and continues at this rate for 3600 simulation seconds. This background traffic flow is created between all nodes in the parent subnet. Thus, the background traffic flows between the nodes LER1-R3 along the two paths LER1 - LSR1 - LSR2 - LSR3 - LER2 - R3 and LER1 - LSR4 - LSR5 - LSR6 - LER2 - R3.

The profile configuration object is another object used to create a user profile named "video." This profile was then specified on different nodes in the network to generate application layer traffic. Sample configuration for the object. The video conferencing starts 150 seconds after the start of the simulation and continues until the end of the simulation. After every 60 seconds, a new video conferencing session will be created between the two workstations.


Profile: Background Traffic

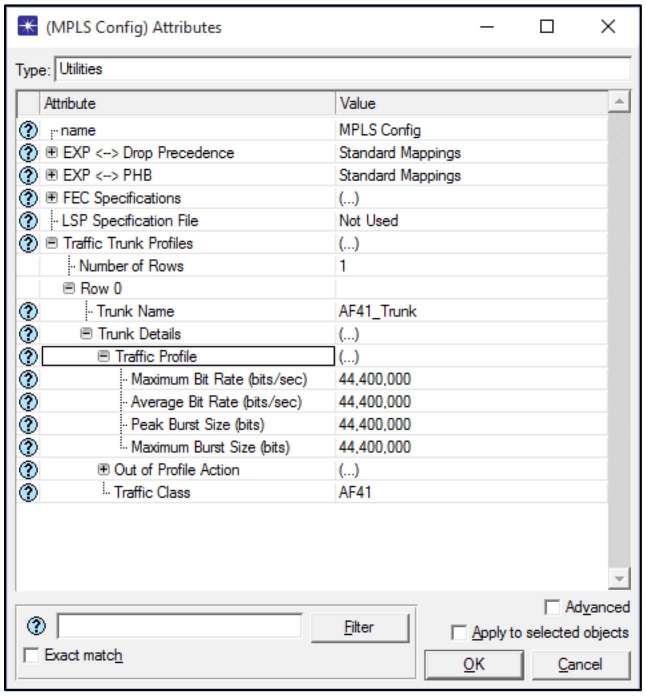
Profile name: Background Traffic

☑ Uniform X intervals: 3,600 seconds/step

☐ Use start time: 21:33:32.000 Nov 14 2015

seconds	bits/second
0.0	40,544,000
3,600	0.0

Figure 9. Traffic flow configuration


(MPLS Config) Attributes

Type: Utilities

Attribute	Value
name	MPLS Config
EXP <-> Drop Precedence	Standard Mappings
EXP <-> PHB	Standard Mappings
FEC Specifications	(...)
LSP Specification File	Not Used
Traffic Trunk Profiles	(...)
Number of Rows	1
Row 0	
Trunk Name	AF41_Trunk
Trunk Details	(...)
Traffic Profile	(...)
Maximum Bit Rate (bits/sec)	44,400,000
Average Bit Rate (bits/sec)	44,400,000
Peak Burst Size (bits)	44,400,000
Maximum Burst Size (bits)	44,400,000
Out of Profile Action	(...)
Traffic Class	AF41

Figure 10. Traffic trunk profile

Finally, MPLS configuration object is used to define the *Forward Equivalency Class* (FEC) and traffic trunk profiles. The FEC defined for the video traffic is the Video_FEC. The *Type of Service* (ToS) or *Differentiated Services Code Point* (DSCP) is set to AF41 for this FEC. Hence, this FEC will be used only by the incoming IP datagram that has the ToS or DSCP field set to AF41. In this study, the video conferencing application has been configured in such a way that the hosts send the IP datagram for the application with DSCP bit set to AF41. Fig. 10 shows the sample configuration for the Trunk Profile.

The delay increased rapidly from 0.5 seconds to 1.05 seconds. After this point, the delay increased linearly and reached 2.3 seconds.

The comparison of delay in video conferencing in MPLS and IP networks using OSPF specifications is shown in Fig. 11. When the video conferencing is done over traditional IP networks, the delay is found to be increasing linearly with time and reached a peak value of 10.6 seconds. When the same video conferencing session is simulated over MPLS network with single LSP, the delay increased gradually to 5 seconds and remained in this range until the end of the simulation. When two LSPs are used for video streaming, there is a significant decrease in the delay. The delay remained under two seconds for the most part of the simulation.

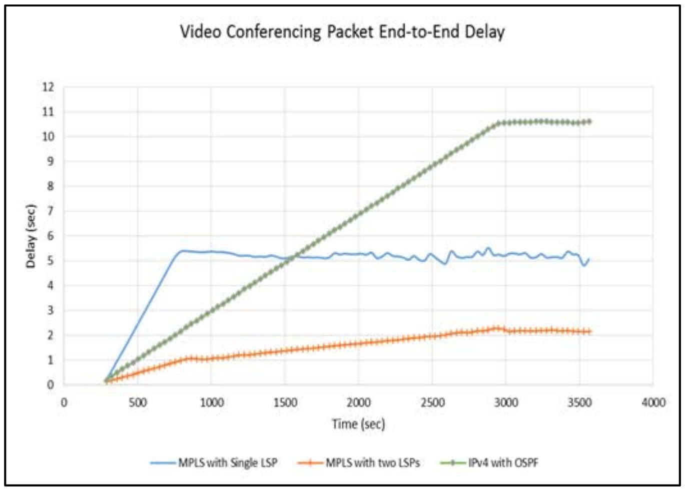


Figure 11. Comparison of end-to-end delay in video conferencing using IP and MPLS

C. *Study on the Location of Multicasting Function*

Consider again the testbed of Fig. 2. In that figure, two groups of 6 destinations are considered in the testbed: group A destinations and group B destinations. Group A destinations is directly connected to the data center network passing through Router1, while group B destination is connected to the data center network after passing through a local router, indicated by "Router," the Internet, and Router2 of the center.

In the first scenario, a video conferencing between two workstations located in San Jose and Bangalore is simulated. The routing protocol used in the scenario is IP using *Open Shortest Path First* (OSPF) specifications and is configured to run on all nodes in the network. Fig. 12 shows the end-to-end packet delay and jitter when video conferencing simulated over the network with the routing protocol OSPF/IP configured on all the nodes. Similarly, Fig. 13 shows the end-to-end packet delay and jitter when video conferencing simulated over the network with the routing protocol OSPF/MPLS configured on all the nodes indicating an improvement over the OSPF/IP case on both delay and jitter.

The Trunk profile used in this paper is AF41_Trunk. The AF41_traffic class is mapped on to this traffic trunk. The maximum bit rate and average bit rate are set to 44,000,000 bits/sec. The maximum burst size and the peak burst size are set to 44,000,000 bits.

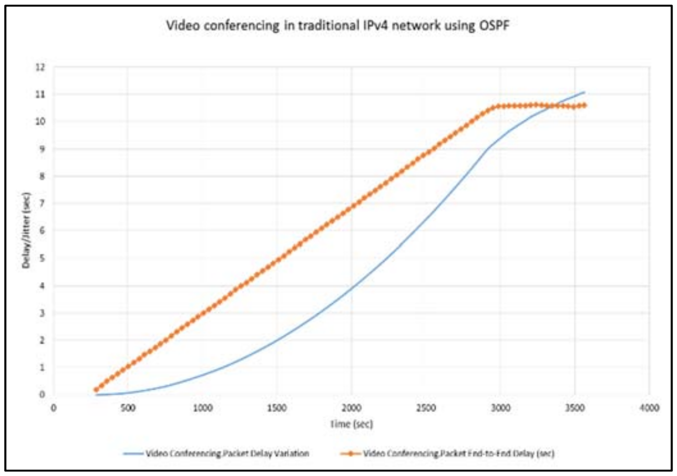


Figure 12. Delay and jitter in video conferencing using OSPF/IP

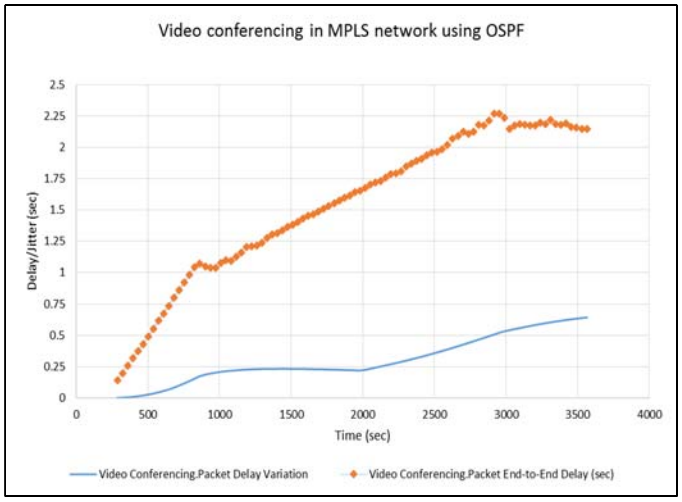


Figure 13. Delay and jitter in video conferencing using OSPF/MPLS

Fig. 14 shows the result of performance evaluation on the throughput for group A destinations. We picked three destinations, Dest. 1, 2, and 3. Then we deployed three different experiments, each considering one of nodes 2, 3, and 5 as the location of the multicasting multimedia streaming. This study clearly shows, that the throughput of multicast at the highest switching nodes (node 5) is highest with an average of 900,000 packets/sec, whereas the one for switching node 2 is the lowest with the average of 110,000 packets/sec. This experiment teaches the multicasting live multimedia must be carried out at the highest levels of the data center switching structures to return the best performance.

The results of the second experiment to determine the best location of multicasting multimedia is shown in Fig. 14. In this experiment, we consider group B destinations in Fig. 3. The study is conducted on Router2 which is attached to the cloud center and Router which is attached to the destinations. The study evidenced by Fig. 15 shows that the queueing delay (QD) at the Router is lower than the one in Router2. What this result teaches us is that the streaming must also be multicast at the closest routing stage to the destinations.

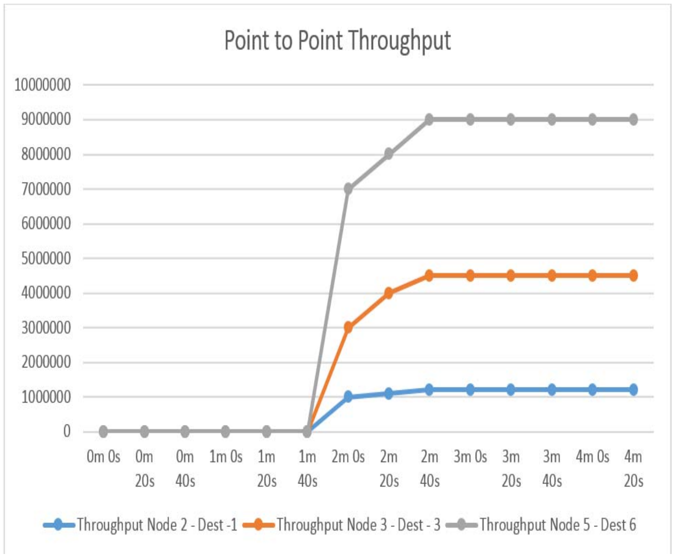


Figure 14. Comparison of throughput at switching nodes 2, 3, and 5

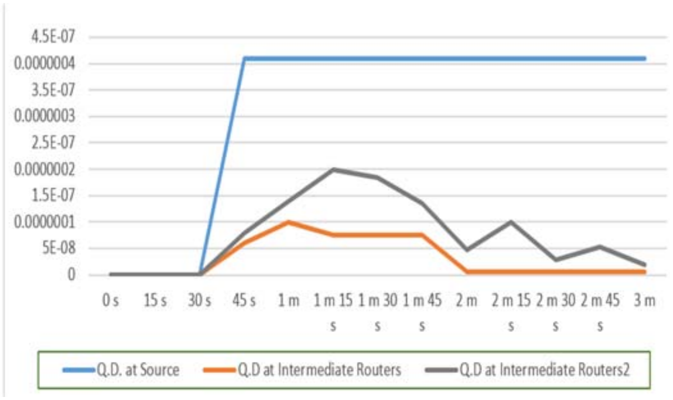


Figure 15. Comparison of queueing delay (QD) at source, and intermediate routers

IV. CONCLUSION

In this paper, we conducted a thorough study on the control of multicasting multimedia streaming over the components of data center networks. We focused on finding the best locations of traffic multicasting, assigning the quality of service (QoS) while multicasting, and the right network devices for this objective. we designed and implemented peer to peer and data center topologies under QoS restrictions and multicast requirement of streaming traffic. The two topologies were implemented for various video streaming applications such as video conferencing and video streaming. The parameters of these video sources were changed in to measure the performance metrics of the multicast networks. Parameters such as video codecs, frame size, frame interarrival rate, link speed, QoS were changed for analysis. From the analysis it was observed that a multitier architecture connected to high speed links was best suited for high end real time traffic. Further it was observed that the QoS configuration for these real time traffic reduces the packet end to end delay and the latency of these packets was also less as compared to other packets. Building a multitier not only helped in better load distribution of traffic across links but also this type of topology was better equipped to handle failures at device, links and server levels. This paper also covered the different multicast routing protocols that can be used. At the end of the paper we analyzed the optimal locations for multicasting multimedia streaming traffic. We learned that the streaming must also be multicast at the closest routing stage to the destinations.

REFERENCES

[1] N. F. Mir, A. T. Rao, and R. Garg, "Efficient Video-Based Packet Multicast Method for Multimedia Control in Cloud Data Center Networks," *Proceedings of 13th International Conference on Systems and Networks Communications* (ICNSC2018), Nice, France, October. 14-18, 2018.

[2] K. Oe, A. Koyama, and L. Batolli, "Performance evaluation of multicast protocols for multimedia traffic," IEEE Xplore document, IEEE 27th International Conference on Advanced Information Networking and Applications (AINA), 2013.

[3] T. Do, K. Hua, and M. Tantaoui, "P2VoD: Providing fault tolerant video-on-demand streaming in peer-to-peer environment." Proc. of the IEEE ICC 2004. Paris: IEEE Communications Society, 2004.

[4] B. Cheng, L. Stein, and J. Hetal, "Grid Cast: Improving peer sharing for P2P VOD," ACM Transactions on Multimedia Computing, Communications, and Applications, vol. 4, 2008.

[5] B. Quinn and K. Almeroth, "IP Multicast Applications: Challenges and Solutions." RFC, 2001, ACM Digital Library.

[6] "OPNET Technologies Inc. OPNET Modeller Product Documentation Release 14.5," OPNET Modeller, 2008.

[7] X. Sun and J. Lu, "The Research in Streaming Media OnDemand Technology based on IP Multicast," 3rd International Conference on Computer and Electrical Engineering, 2010.

[8] http://www2.ic.uff.br/~michael/kr1999/4-network/4_08-mcast.htm, Access date: April 2019.

[9] Z. Bojkovic, B. Bakmaz, and M. Bakmaz, "Multimedia Traffic in New Generation Networks: Requirements," Journal of Control and Modeling, 2016.

[10] S. Khanvilkar, F. Bashir, D. Schonfeld, and A. Khokhar, "Multimedia Networks and Communication," 2015

[11] X. Wang, C. Yu, and H. Schulzrinne, "IP Multicast Simulation in OPNET," Semantic Scholar, 2015.

[12] N. Mir, "Computer and Communication Networks," Pearson Prentice Hall, 2nd Edition, 2014.

.

Object Recognition Using Neural Networks and Complex Reflection Signals

Short paper

Hristomir Yordanov* and Irina Topalova†
Faculty of German Engineering Education and Industrial Management
Technical University of Sofia, Sofia, Bulgaria
Emails: *yordanov@fdiba.tu-sofia.bg, †itopalova@abv.bg

***Abstract*—Radar based imaging techniques can be used to collect 3D information about objects, which in turn can be used to identify and measure specific parameters of these objects. Such measurements need to correlate specific radar signals with the object properties. This can be done using neural networks, as they are designed to search for patterns, which are difficult to find using analytic methods. This work presents a neural network based reflection signal processing system for object identification by attempting to identify an object placed in a rectangular waveguide. We extract both the phase and the amplitude of the reflected signal and compare recognition systems using amplitude only and using both phase and amplitude.**

Keywords—Scattering signals; Object identification; Neural Networks

I. Introduction

Radars find many applications as imaging tools. Using the property of electromagnetic waves to partially penetrate and partially reflect from dielectric materials, they can provide 3D images of a large set of objects. The development of easily available high-frequency components up in the microwave, millimeter wave and even terahertz ranges allows for high spatial resolution of the obtained images. This technique finds multiple applications in security systems, in medical systems and in agriculture.

We can consider as an example the sensor described in [2]. The system consists of a 24 GHz Frequency Modulated Continuous Wave (FMCW) radar used to make 3D images of grapevine plants in order to estimate the volume of grapes in a given plant. The radar is equipped with a high gain antenna and is mounted on a pan-tilt platform, which allows for performing azimuthal and elevation scans. The radar bandwidth is 2 GHz. This setup allows for a $7.5\,\mathrm{cm}$ depth resolution (that is the precision of the measurement of the distance between the object and the radar) and transverse resolution of $1.5\,\mathrm{cm}$. Of course, using higher signal frequency, bandwidth and more directive antennas, resolutions in the millimeter range can be achieved [3].

The processing of the radar signal in order to obtain information about the object parameters of interest can be a challenging task. The measurement system described in [2] relies on statistical analysis in order to obtain the grapes volume. Neural networks are optimized for pattern search in complex data. Therefore, they can be used in radar based measurement systems as they can extract the data of interest from the clutter and simultaneously estimate the value of the parameter of interest. In the grapevines radar example, the parameter of interest is the volume of the produced grapes and the clutter is the signals from the plant's trunk and leaves.

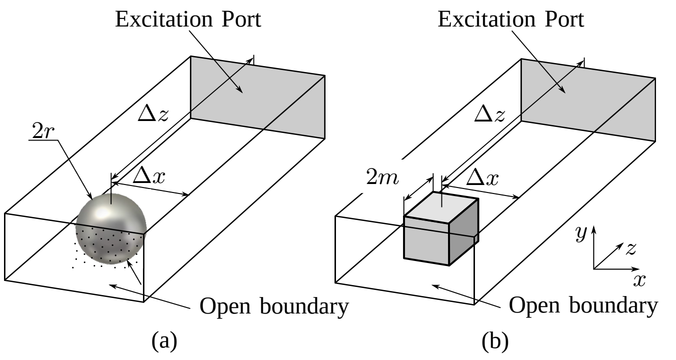


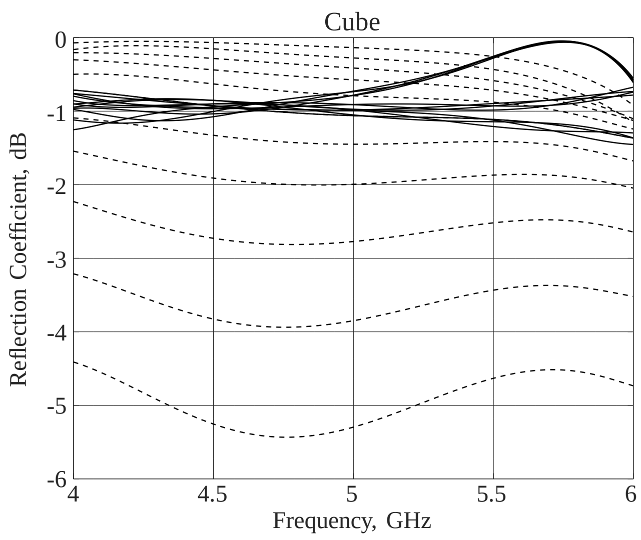
Fig. 1. Positioning of a ball of a diameter $2r$ (a) and a cube with edge $2m$ (b) in a WG-12 rectangular waveguide.

In order to develop an intelligent 3D image processing system, we need to start by implementing simple 1D solutions. In this paper, we present a neural network for shape recognition based on the scattered signal as a benchmark case study. The investigated object is a body of perfectly conducting material placed in a rectangular hollow waveguide. This limits the neural network input signal to the spectral representation of a single point reflection signal. We have previously presented identifying simple objects using just the amplitude of the reflected signal [1]. In this paper we consider both the amplitude and the phase of the scattered wave. The setup has been modeled numerically and the scattering paarmeters have been obtained using computer simulation.

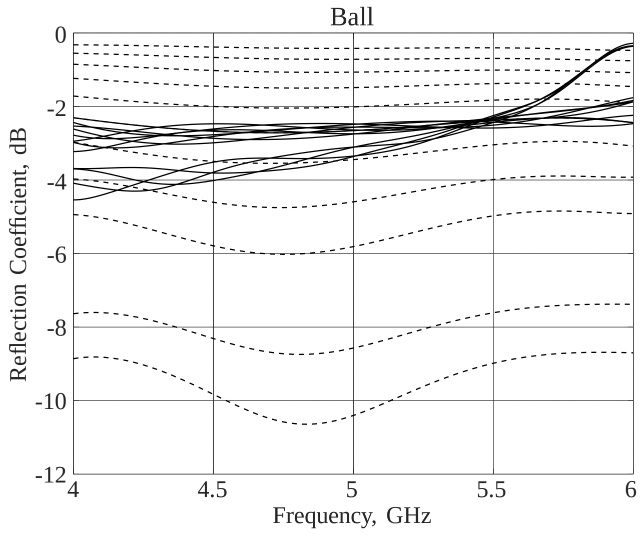
Section II describes the setup of the performed simulation and shows the computed reflected signals from the two types of objects in a waveguide. Section III details the neural network based signal processing used to identify the objects based on the reflected signal. Section II discusses the obtained experimental results and Section V summarizes the paper and sketches the future work.

II. Experimental Setup

The experimental setup consists of a hollow rectangular WG12 waveguide with an object placed at distance Δz form the excitation port, as shown in Figure 1. The cross-sectional dimensions of the waveguide are $a = 47.5\,\mathrm{mm}$

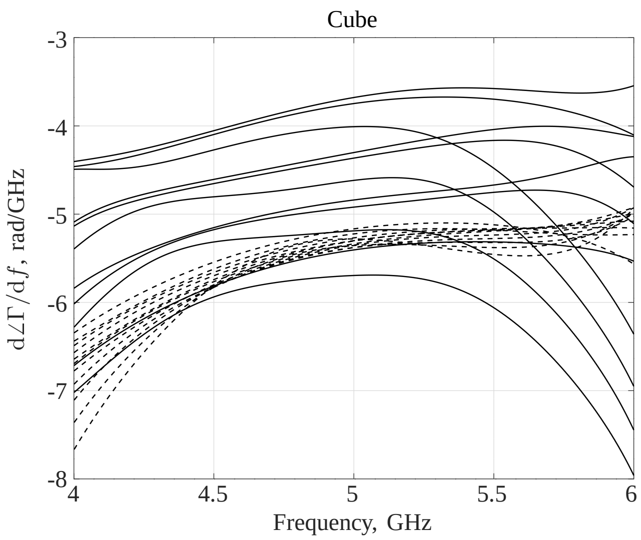


(a)

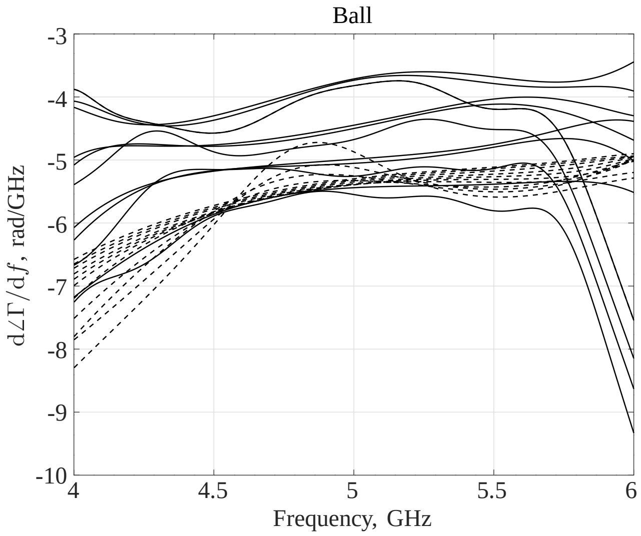


(b)

Fig. 2. Family of curves showing the magnitude of the reflection coefficient of a waveguide with a conducting cube (a) and a sphere (b) inside. The solid lines and the dotted lines represent varying position and size of the sphere respectively.



(a)



(b)

Fig. 3. Family of curves showing the derivative of the phase with respect to frequency of the reflection coefficient of a waveguide with a conducting cube (a) and a sphere (b) inside. The solid lines and the dotted lines represent varying position and size of the sphere respectively.

and $b = 22.1\,\text{mm}$. The scattering objects are a sphere of radius r (Figure 1a) and a cube of length $2m$ (Figure 1b). Both objects are made of a perfect electric conductor and are placed at a distance of Δx from the short wall of the conductor. The objects were placed in the middle of the waveguide in the vertical y direction. The excitation port has been placed at the $-z$ end of the waveguide. The opposite end has been terminated with an open boundary in order to model an infinitely extended waveguide and thus eliminate the reflections from that boundary. The model has been simulated for the frequency range of 4 to 6 GHz, which corresponds to the full single mode range of the waveguide. We measure the reflection coefficient at the excitation port. The used simulation tool is CST Microwave studio.

Two families of results have been generated. First, we varied the dimensions of the objects—the sphere radius r and the cube edge $2m$ respectively—while keeping both objects at fixed position $\Delta z = 100\,\text{mm}$ and $\Delta x = a/2$, that is 100 mm from the excitation port and in the middle along the x direction. The size parameters r and m varied from 4 to 10 mm in 0.6 mm steps. Then, we held the object dimensions fixed at $r, m = 7\,\text{mm}$ and varied the offset dimension as follows:

$$\Delta z = 0 \text{ to } -30\,\text{mm in } 10\,\text{mm steps,}$$
$$\Delta x = 0 \text{ to } 10\,\text{mm in } 5\,\text{mm steps.}$$

The full combination of offset coefficients has been modeled.

We consider both the amplitude and the phase of the reflected signal, relative to the incident one. This is defined as the complex reflection coefficient Γ of the perturbed waveguide [4]. The amplitude and the phase are two independent variables and we can get more information about the shape of the object in the waveguide if we consider both of them instead of just one. The information carried by the distribution of the amplitude of the reflection coefficient in frequency $\Gamma(f)$ can be extracted straightforward by feeding it directly to a neural network, as we proceed in Section III. There is an intrinsic difficulty in working with the phase, though, because we can not distinguish a 2π phase increment: $\angle\Gamma = \angle\Gamma \pm n2\pi$. In other words, the reflection coefficient generated by a perfectly conducting transverse wall, shorting a lossless waveguide, will be the same as the one when the wall is moved $\pm\lambda_g/2$ in longitudinal direction, where λ_g is the length of the guided wave. This can cause significant difficulties for an intelligent system, trying to identify the shape of the object irrespective of the distance of interrogation.

We attempt to circumvent this problem by using the derivative of the phase of the reflection coefficient with respect to the frequency $\mathrm{d}\angle\Gamma/\mathrm{d}f$, measured in rad/Hz, instead of the phase itself. In this way we disregard any $\pm n2\pi$ uncertainty while keeping the information about the distribution of the phase in frequency.

The results for the amplitude of the reflection coefficient for a cube and a ball are presented in Figures 2a and 2b, respectively, where the dotted lines show the family of curves for varying object size, while the position is held fixed, and the solid lines show the results for fixed size and varying offset. The dotted lines show a greater reflection coefficient ans the object dimensions r and m increase, which can be expected as larger objects create larger echo. The derivative of the phase with respect to frequency for a cube and a ball is presented in Figures 3a and 3b respectively.

We have used a combination of the frequency distribution of the magnitude and phase of the reflection coefficient in order to generate an input for the shape recognition neural network. We have used 11 points from each of the curves from Figures 2 and 3, as the frequency response varies slowly and using this representation we lose no information. Thus we get an input signal of 22 points for each size and position of the respective object. As the number of size and position varying simulations is also 22, we get 22 input signals of 22 points each for each object. We use 14 of those signals to train the network and 8 to test it.

We compare the efficiency of a shape recognizing neural network working with reflection coefficient amplitude and phase versus a system working with amplitude only. We use 11 points from each amplitude curve, presented in Figure 2 in order to train and test such a network.

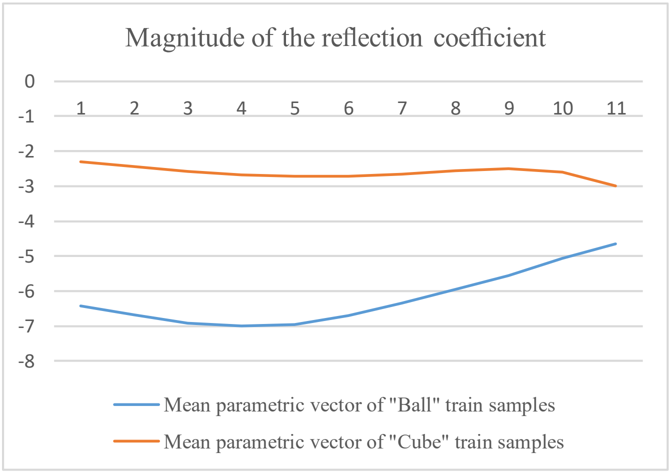


Fig. 4. Mean parametric vectors over the magnitudes of the reflection coefficients for 14 train samples of "ball" and "cube".

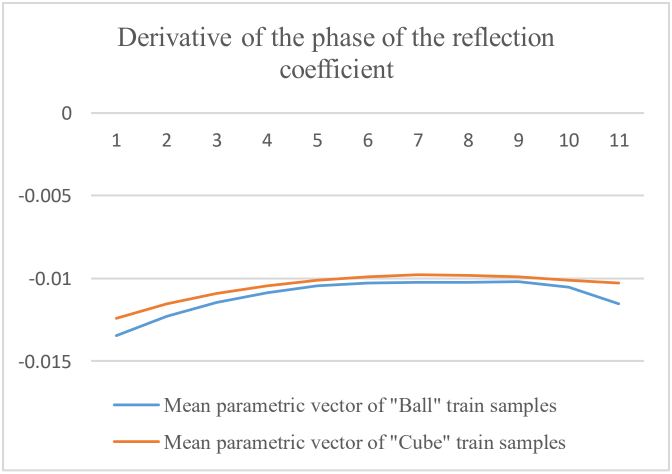


Fig. 5. Mean parametric vectors over the derivative of the phase of the reflection coefficients for 14 train samples of "ball" and "cube".

III. Neural Network Signal Processing

There are various studies concerning the pre-processing of input data for the recognition system to improve its efficiency and accuracy. In order to make a correct choice of the recognition method, it is necessary to analyze the data defining the parametric descriptions of the objects. This analysis is based on the calculation of the statistical parameters of the data as well as the determination of the degree of similarity between the parametric descriptions of the objects. Since our study involves highly correlated input parametric vectors representing parametric descriptions of the reflected radar signal for the two objects under study, we have chosen the adaptive neural network method, that provides the most effective recognition of similar input data. As the two 3D objects have similar shapes, it is necessary to use an adaptive and precise method for recognition and classification of the two objects. The Deep Learning method using a Multi-Layered-Perception (MLP) feed forward Neural Network (NN), trained by the Back-propagation (BP) algorithm, gives satisfactory results in the cases described in [5], [6]. This allows precise placement of boundaries between object classes with overlapping parametric descriptions – in our case very similar reflection signals. In order for the neural network to be "assisted" in advance, different linear transformations (mostly scaling to the ranges of $(0, 1)$ or $(-1, 1)$) [7], [8], statistical standardization (using deviation from the mean) or various other appropriate mathematical transformations over the input data [9], [10] are suggested. In our study, in order to reduce the preliminary calculations and simplify the method, we choose an appropriate combination of independent parameters of the reflected radar signal, such as magnitude and phase of the reflection coefficient.

TABLE I. STANDARD DEVIATION AND CORRELATION BETWEEN THE MEAN PARAMETRIC VECTORS

Input data	Standard deviation		Correlation between mean parametric vectors
	Ball	Cube	
$\lvert\Gamma\rvert$	0.801	0.179	-0.337
$\mathrm{d}\angle\Gamma/\mathrm{d}f$	$1.05\cdot10^{-3}$	$8.29\cdot10^{-4}$	0.972
Γ	3.217	1.337	0.976

A. Preprocessing stage

In this stage some statistical parameters of the signals are calculated, in order to evaluate the correlation between the signals representing the two objects and the mean square deviation of the signal parameters concerning the training samples for each of the two objects. For this purpose, the mean parametric vectors of the magnitude, of the derivative of the phase of the reflection coefficient and of the complex signal (combining both of them) are calculated. The obtained mean parametric vectors for 14 train samples of "ball" and "cube" are shown in Figures 4 and 5, respectively.

The next step is to evaluate the standard deviation for each of these two signals and calculate the correlation between the mean parametric vectors of "ball" and "cube". Considering these two parameters, it is easier to make decision what kind of a recognition method to apply, since with a high correlation of interclass parametric descriptions, it is recommended to choose an adaptive recognition method, such as a neural network.

On the other hand, the adaptation of the neural network and, respectively, the accuracy of recognition in this case would be much more efficient, if the standard mean square deviation of the input training parameter vectors within the class is higher. The correlation between the mean parametric vectors over the magnitudes, the derivative of the phase of the reflection coefficients and over the complex (magnitude and derivative of the phase) signal for 14 training samples of "ball" and "cube" respectively, is has been calculated using the Pearson correlation coefficient [11]:

$$\rho_{\text{ball,cube}} = \frac{\sum_{i=1}^{n}(B_i-\overline{B})(C_i-\overline{C})}{\sqrt{\sum_{i=1}^{n}(B_i-\overline{B})^2(C_i-\overline{C})^2}}, \qquad (1)$$

where B_i, C_i is the current component of the input vector "cube/ball", and n is the number of components ($n = 11$ for input vector "magnitude" and "derivative of the phase"; $n = 22$ for the "complex" vector). The achieved results for the discussed calculated parameters are shown in Table I, where Γ is the complex reflected signal, $|\Gamma|$ is its magnitude, and $\mathrm{d}\angle\Gamma/\mathrm{d}f$ is the derivative if the phase of the reflection signal with respect to the frequency. The obtained results show that the correlation has the lowest values for input data "magnitude", but the standard deviation is highest at the complex signal. Thus we will train a MLP neural network (MLP NN) with "magnitude" and "complex" signals, aiming to compare the recognition accuracy results.

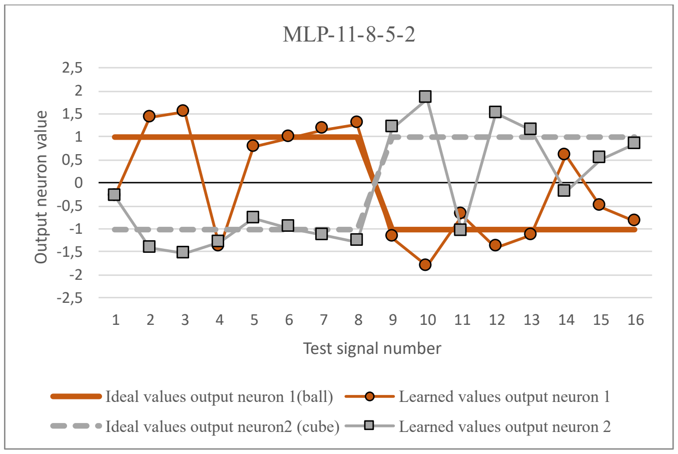


Fig. 6. MLP NN (11-8-5-2) output results for Output neuron 1 (ball) and for Output neuron 2 (cube) when recognizing the 8 exemplars of objects ball and cube with "magnitude" input vector

B. Neural network: structure and training method

We have trained the MLP NN in two cases: first only with "magnitude" input signal, sampling 11 points from each curve, as they vary slowly in frequency and second with "complex" signal having 22 points respectively. In both cases the training set contains 14 curves with varying offset and object size. The test set contains 8 specimens, representing the two types of objects, whose reflected signals have not participated in the training set. For the first case we have designed the MLP NN by adding two hidden layers and increasing the number of neurons in each layer until satisfactory recognition was achieved. The best recognition results were obtained in the case MLP 11-8-5-2 structure (with two hidden layers, having 8 and 5 neurons and 2 output neurons, representing the two recognizable objects), with a reached minimum Mean Square Error (MSE-ε) of 5%. For the second case we train different MLP structures of 22-10-2; 22-15-2 and 22-20-2 neurons with a reached a few times less minimum MSE–ε) of 0.02 and 0.04%.

In both cases a step by step "continue" stage of the training has been applied, reducing the error achieved and accepted at each previous stage. We use steps obtaining MSE of 5%; 1%; 0.8%; 0.4%; 0.1%; 0.08%; 0.04%; 0.02%. This method permits fine tuning (FT) of a pre-trained network using slightly changed training data.

IV. EXPERIMENTAL RESULTS

The MLP NN output results when recognizing the 8 specimens, representing the two objects, whose reflected signals have not participated in the training set, are shown in Figures 6 to 10. Figure 6 represents the NN outputs 1 and 2, when training the network only with "magnitude" input data for a 11-8-5-2 MLP NN. In this case the obtained recognition accuracy is 75% for both objects, that is, 2 samples of each object are falsely recognized. The training iterations were stopped when the MSE has reached 5%. Figures 7, 8 and 9 show the NN outputs 1 and 2, when training the network with "complex" input data, respectively with different MLP NN structures: 22-10-2; 22-15-2 and 22-20-2. In order to put

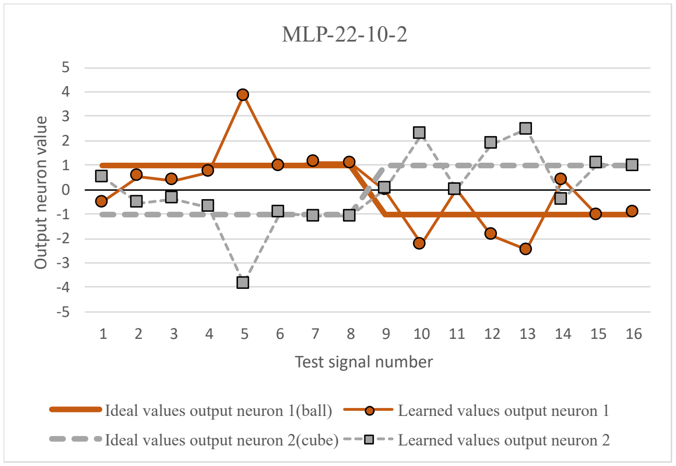


Fig. 7. MLP NN (22-10-2) output results for Output neuron 1 (ball) and for Output neuron 2 (cube) when recognizing the 8 exemplars of objects ball and cube with “complex” input vector

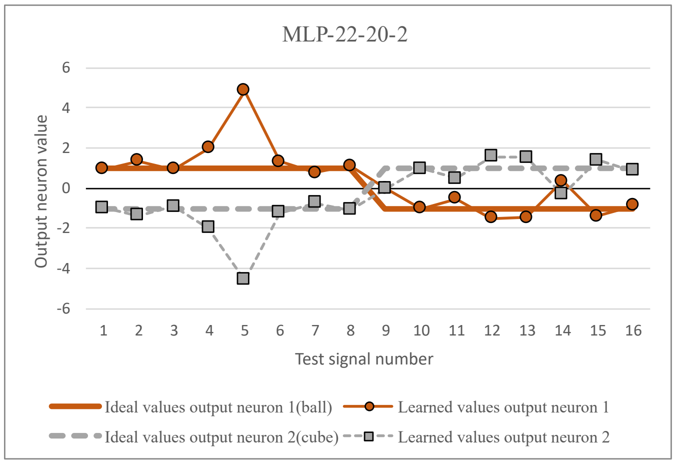


Fig. 9. MLP NN (22-20-2) output results for Output neuron 1 (ball) and for Output neuron 2 (cube) when recognizing the 8 exemplars of objects ball and cube with “complex” input vector

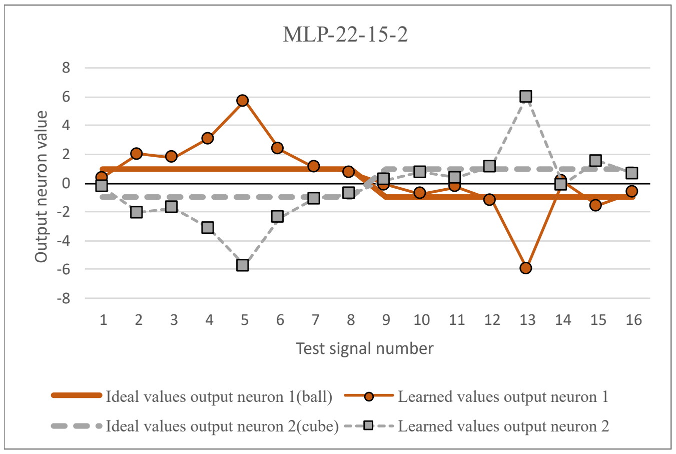


Fig. 8. MLP NN (22-15-2) output results for Output neuron 1 (ball) and for Output neuron 2 (cube) when recognizing the 8 exemplars of objects ball and cube with “complex” input vector

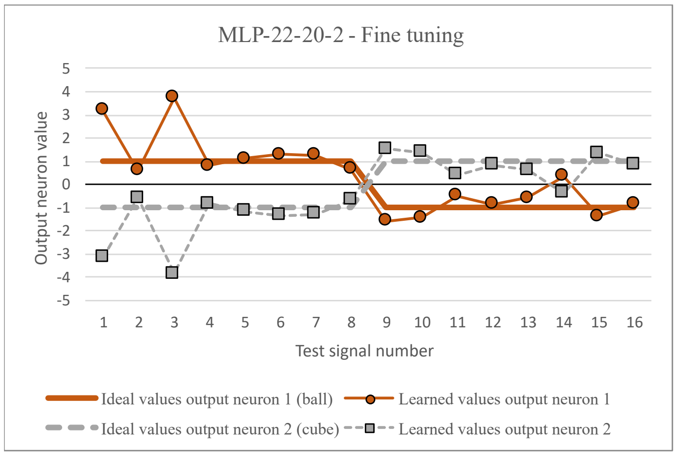


Fig. 10. MLP NN (22-20-2) output results for Output neuron 1 (ball) and for Output neuron 2 (cube) when recognizing the 8 exemplars of objects ball and cube with “complex” input vector applying fine tuning training

more precise boundaries between the object classes and to improve the accuracy of recognition, it is necessary to increase the number of neurons in the hidden layer of the MLP NN. Thus, each subsequent train and test step is made with an increased number of neurons in the hidden layer. It is good recognizable that the approximation of ideal/ learned values is improved after each subsequent increase of neurons in the hidden layer. For object “ball” the accuracy increases from 87.5% to 100% and for “cube” – from 62.5% (3 samples out of 8 are misidentified) to 87.5% (1 sample out of 8 is misidentified). Figure 10 represents the NN outputs in test phase, when fine tuning train method was applied. The obtained approximation error for the various structures is shown in Figure 11. Obviously, the best approximation was achieved in the case of MLP 22-20-2 and fine tuning training. The summary of the achieved recognition accuracy and the reached MSE for all tested cases, is shown in Table II.

V. Conclusion

This paper shows the initial work on identifying suitable neural network signal processing tools for radar based shape recognition techniques. The achieved recognition results show that it is very appropriate to implement MLP NN for 3D object recognition, when using radar reflection signals. The good approximation abilities of the MLP NNs make it possible to recognize even objects of very similar shapes. It has been shown that a complex signal that has a higher value for standard deviation, results in effective training and therefore in better recognition accuracy. As future work, we intend to test the method for a larger number of objects with similar 3D

TABLE II. Recognition Accuracy for Different MLP Structures and Input Signals

Input Data	MLP structure	Recognition Accuracy, %		
		Ball	Cube	MSE-ε
Magnitude	11-8-5-2	75%	75%	5%
Complex	22-10-2	87.5%	62.5%	0.02%
Complex	22-15-2	100%	75%	0.02%
Complex	22-20-2	100%	87.5%	0.04%

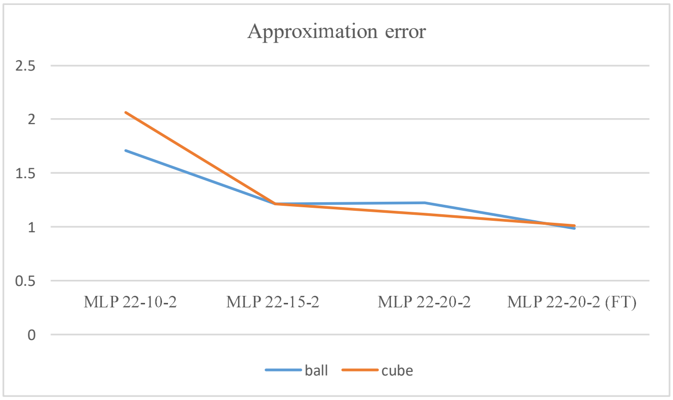


Fig. 11. Achieved approximation error for "complex" input vector with different MLP NN structures.

object shapes. Also, to generalize the method, the test sample set will be increased. Additional calculations of approximation error are also foreseen. The presented results provide shape recognition by a single point wideband reflected signal, which is a model of a pulse radar. We intend to expand these results toward scanning pulsed and scanning frequency modulated continuous wave (FMCW) radars.

References

[1] H. Yordanov and I. Topalova, "Neural networks for scattering signal based object recognition," in *The Fourteenth International Conference on Autonomic and Autonomous Systems ICAS 2018*, May 2018, pp. 1–3.

[2] D. Henry, H. Aubert, T. Veronese, and E. Serrano, "Remote estimation of intra-parcel grape quantity from three-dimensional imagery technique using ground-based microwave FMCW radar," *IEEE Instrumentation Measurement Magazine*, vol. 20, no. 3, pp. 20–24, June 2017.

[3] K. B. Cooper, R. J. Dengler, N. Llombart, T. Bryllert, G. Chattopadhyay, E. Schlecht, J. Gill, C. Lee, A. Skalare, I. Mehdi, and P. H. Siegel, "Penetrating 3-d imaging at 4- and 25-m range using a submillimeter-wave radar," *IEEE Transactions on Microwave Theory and Techniques*, vol. 56, no. 12, pp. 2771–2778, Dec 2008.

[4] D. M. Pozar, *Microwave Engineering*, 3rd ed. Ney York, NY: John Wiley & Sons, 2005.

[5] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608014002135

[6] *NeuroSystems V4.0 Manuall*. Germany: Siemens GmbH, 2016.

[7] N. M. Nawi, W. H. Atomi, and M. Rehman, "The effect of data pre-processing on optimized training of artificial neural networks," *Procedia Technology*, vol. 11, pp. 32–39, 2013, 4th International Conference on Electrical Engineering and Informatics, ICEEI 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2212017313003137

[8] K. Kuźniar and M. Zajac, "Some methods of pre-processing input data for neural networks," *Computer Assisted Methods in Engineering and Science*, vol. 22, no. 2, pp. 141–151, 2017. [Online]. Available: http://cames.ippt.gov.pl/index.php/cames/article/view/33

[9] I. Topalova and M. Uzunova, "Recognition of similar marble textures through different neural networks with de-correlated input data," *International Journal on Advances in Intelligent Systems*, vol. 10, pp. 474–479, 2017. [Online]. Available: http://www.iariajournals.org/intelligent_systems/

[10] I. Topalova, "Automated marble plate classification system based on different neural network input training sets and plc implementation," *International Journal of Advanced Research in Artificial Intelligence*, vol. 1, no. 2, pp. 50–56, 2012. [Online]. Available: http://dx.doi.org/10.14569/IJARAI.2012.010209

[11] J. L. Rodgers and W. A. Nicewander, "Thirteen ways to look at the correlation coefficient," *The American Statistician*, vol. 42, no. 1, pp. 59–66, 1988. [Online]. Available: https://doi.org/10.1080/00031305.1988.10475524

www.iariajournals.org

International Journal On Advances in Intelligent Systems
issn: 1942-2679

International Journal On Advances in Internet Technology
issn: 1942-2652

International Journal On Advances in Life Sciences
issn: 1942-2660

International Journal On Advances in Networks and Services
issn: 1942-2644

International Journal On Advances in Security
issn: 1942-2636

International Journal On Advances in Software
issn: 1942-2628

International Journal On Advances in Systems and Measurements
issn: 1942-261x

International Journal On Advances in Telecommunications
issn: 1942-2601