Optimization of Network Monitoring

Ivana Palúchová[*]
Institute of Computer Engineering and Applied Informatics
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava
Ilkovičova 2, 842 16 Bratislava, Slovakia
ivana.paluchova@stuba.sk

Abstract

The monitoring of a computer network is often challenging. On one hand, monitoring of every node and link gives us all the needed information. On the other hand, this approach generates huge amounts of data that need to be collected, analyzed and processed. In our work, we propose a monitoring model based on a selection of critical nodes in the network. The network monitoring can be applied only on a subset of network elements and therefore reduce the amounts of monitored, collected and processed data. Our proposed solution was verified in a simulation environment of Matlab R2018b. The testing shows that with monitoring of 64% of all nodes in the network we are able to gain 86.7% knowledge of all network elements. In terms of artificial traffic, we proved the reduction of 15.7% - 57.4% of bandwidth consumption in simulated SDN topologies.

Categories and Subject Descriptors

C.2.0 [**Networks**]: General; C.2.1 [**Networks**]: Network Architecture and Design; C.2.3 [**Networks**]: Network Operations

Keywords

network monitoring, performance monitoring, critical nodes, monitoring overhead, software-defined network

1. Introduction

Accurate knowledge of the status of the network is the most vital information for a network administrator. Up-to-date information about the traffic load, performance parameters or potential problems is crucial for everyday operation of the network. Monitoring of multiple network parameters on every link or node might create an issue, mainly in large and dense networks [1]. Time needed for measurement, collection and analysis of data may be too long compared to rapidly changing network conditions. The results might be outdated and selected reaction not adequate [2].

The new concept of networking brought by SDN changed also the approach to network monitoring [3]. The standard monitoring solutions designed for traditional IP networks are often not flexible enough to cover new opportunities of SDN architecture [4]. With the expansion of network scale grows the amount of data needed to be measured, collected and analyzed. In the context of SDN the performance of the controller itself can become the bottleneck of the requirements [5].

In our work we focus on problems in performance monitoring in large networks – both traditional IP networks and SDNs. Our goal is to propose a novel monitoring model, which could provide necessary information about the network status, while lowering the monitoring overhead and computational complexity. With the approach of selecting only important nodes for the monitoring we propose a solution to lower the amount of artificial traffic needed for monitoring and therefore fasten the collection, analysis and decision making.

The structure of this extended abstract is as follows. We present existing monitoring solutions for traditional IP networks as well as SDNs in the second section. The third section is dedicated to the proposed solution and used methodology. The evaluation and testing results are described in the fourth section. Last fifth section contains the conclusions of our work.

2. Related Work

In the following sections we provide an overview of the current state of the art in the area of network monitoring in general and in the area of SDN monitoring. The problems and complications known in traditional IP network monitoring appear also in the monitoring of SDNs. Furthermore, new challenges are rising in SDN, where the solutions and approaches need to be adjusted due to different network architecture and principles.

All of the analyzed proposals to network monitoring have one common feature – the need to find a balance between monitoring all details in the whole network and keeping the monitoring overhead low while obtaining necessary information about the network status. Applying moni-

[*] Recommended by thesis supervisor: Prof. Pavel Čičák
Defended at Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava on June 26, 2020.

Palúchová, I. Optimization of Network Monitoring. Information Sciences and Technologies Bulletin of the ACM Slovakia, Vol. 12, No. 1 (2020) 7-12

toring to every network element may become an issue, mainly in large networks. Increased CPU overhead on the monitoring nodes, higher bandwidth utilization and huge amounts of monitored data increase the time and costs for the monitoring [6, 7].

2.1 Traditional IP Network Monitoring

We can distinguish two main categories of network monitoring – availability monitoring and performance monitoring. The first is concerned with the accessibility of the network devices such as application servers while the second type of monitoring deals with the network performance parameters on the network links and nodes, such as available bandwidth, delay, jitter or packet loss.

For both types of monitoring there is number of existing solutions on the market, such as [8, 9, 10, 11, 12]. These systems differ in depth of the analysis, features and graphical representations of the monitored data. Along with these ”standard” solutions there are many new approaches being proposed by the academics.

In the area of availability monitoring researchers address various challenges such as heterogeneity and complexity in large size industrial networks, inefficiency of troubleshooting or need for a centralized monitoring solution. Authors in [13] proposed a novel and flexible monitoring solutions based on existing Nagios software. Authors in [14] focused on the collection and visualization of important information that characterizes the functionality and operating conditions of the network and network services.

The monitoring of device performance is crucial in the network along with the status and health monitoring. The question of performance monitoring is widely analyzed in the academic field, covering different areas of application. The common challenge, however, still remains - the possibility to monitor the network effectively, accurately and flexibly.

The question of QoS monitoring was analyzed and new approaches were proposed by authors in [15, 16]. Another area of interest is focused on monitoring of traffic flows in the network. Authors in [17] address the problem of local traffic monitoring overloads, publications [18, 19] proposed a parallel monitoring of flows to measure end-to-end delay in the network.

2.2 SDN Network Monitoring

The new concept of networking brought by SDN changed also the approach to network monitoring. The SDN architecture provides new possibilities to measure various network parameters, monitor link failures or topology changes. OpenFlow managed nodes usually report their status periodically to the SDN Controller. Furthermore, the Controller can request specific information such as flow statistics, interface statistics, etc.

The standard monitoring solutions designed for traditional IP networks are often not flexible enough to cover new opportunities of SDN architecture. This drives a huge number of academics into research in this area.

Authors in many publications addressed the challenges of QoS monitoring in SDNs: [20] proposed a passive approach to bandwidth measurement utilizing existing OpenFlow messages; [4] analyzed the possibilities of latency monitoring and potential bottleneck of the SDN controller, while proposing an optimization algorithm to reduce the time needed for delay measurements; [21] proposed a two-way link-level packet loss measurement solution using active measurement.

Traffic flow monitoring in SDN is challenging as it requires to install an entry per flow in the flow tables of every switch and the controller might become a bottleneck. Many researchers focused on this area and proposed novel solutions: [22] proposed a flow monitoring solution with classification where the measurements are maintained in the switches and are sent to the controller asynchronously; an interesting flow monitoring scheme is described in [23] where a new polling mechanism called poll-some is introduced. As compared to the two existing mechanisms, poll-single and poll-all, poll-some aims at collecting the statistics of multiple not-yet-covered flows at a switch.

Monitoring a large number of nodes can generate unsustainable loads on the central controller due to the increasing amount of measurement traffic converging to it [24]. To overcome this problem authors proposed number of different distributed solutions of the network monitoring: a distributed controller scenario was proposed in [25] which tries to minimize the measurement overhead, message interactions and CPU utilization on the SDN controller; in [24] the authors introduced a configurable interface for the synchronization of monitoring data between local managers operating within a distributed management environment.

3. Proposed Solution

The goal of our work was to design a new network monitoring model which will select the key network elements and identify their relationship with the rest of the network. The network monitoring can be then focused on these identified parts of the network instead of the whole network. In the end, the proposed model will provide information about all network elements without the need of extensive network monitoring. In result the proposed solution will cause less monitoring overhead, while still receive all needed information about the network status. Less bandwidth usage and less monitored data to be analyzed will provide faster decision making for the network manager and more available bandwidth for the end users.

An overview of the proposed model structure is shown in Figure 1. The primary input for the model consists of:

- network topology which models the real network
- user input such as type of NPPs (network performance parameters) to monitor

The primary input will be used for the calculations in our model. The calculations will produce following primary outputs which are also used as secondary inputs for the proposed model:

- identification of the important elements which should be monitored
- identification of the influences among network elements

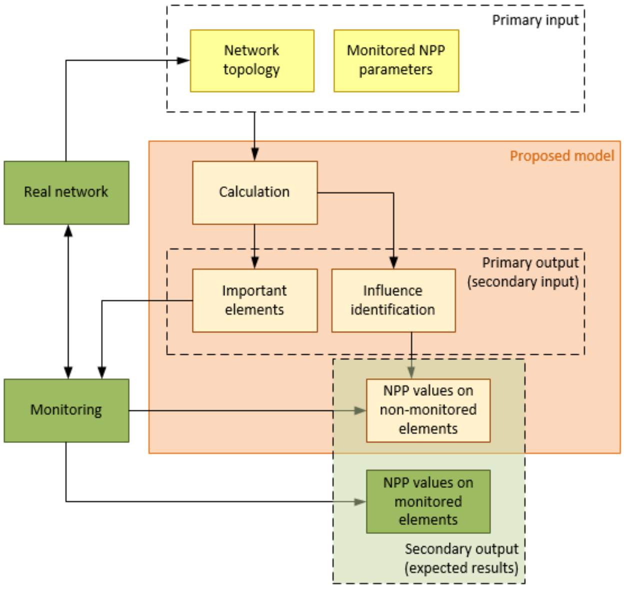

Figure 1: Proposed model structure.

The monitoring probes will be set to the identified important elements. Values obtained by monitoring together with the identified influence will result into NPP values estimation on elements which are not monitored.

3.1 Selection of Critical Nodes

The proposed approach lays in identifying set of *critical nodes* which will serve as main sources of monitoring data in the network. Overall network status will be calculated and estimated based on information gathered from these *critical nodes*. The identification of *critical nodes* is based on weights we assign to each node and link. The weight of node n_p is defined in Equation 1 and it is statically assigned based on the status of the node. The weight of link l_{ij} is defined in Equation 2 and it is dependent on the weights of nodes a, b which are connected by this link where node a represents the source node and node b represents the destination node of the link.

$$w_{n_p} = \begin{cases} 1 & \text{for critical node} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$w_{l_{ij}} = w_{n_a} + w_{n_b}/2 \tag{2}$$

The goals for the selection of *critical nodes* are:

- minimize the number of selected *critical nodes*
- maximize the number of links with *full knowledge*
- cover all links in the network (no link can be left unmonitored)

Although an optimal solution for *critical nodes* selection can be found easily for small networks, it can be, however, a serious time-consuming and resource-demanding task for larger networks. Therefore, we formulate this problem as a multi-objective optimization problem consisting of two objectives as described in Equation 3 and Equation 4 and one constraint described in Equation 5.

$$min \sum_{p=1}^{S} w_{n_p} subject \begin{cases} \forall w_n & \text{are integer} \\ \forall n \in N \end{cases} \tag{3}$$

$$max \sum_{i,j=1}^{S} w_{l_{ij}} subject \begin{cases} \forall w_l & \text{are integer} \\ \forall l \in L \end{cases} \tag{4}$$

$$\forall l \in L : w_l > 0 \tag{5}$$

We used Matlab integer linear programing (*intlinprog*) function to solve the multi-objective optimization problem. The function as described in [26] finds the minimum of a problem specified by Equation 6 where f, x, *intcon*, b, *beq*, *lb*, and *ub* are vectors, and A and *Aeq* are matrices:

- f represents the objective function – sum of all node's weights as described in Equation 3
- x represents the decision variable – node's weight w_n
- *intcon* specifies the components of x that are integer – all of the nodes are integer variables
- b represents the constant vector for the inequality constraint – described in Equation 7
- *beq* represents the constant vector for the equality constraints – not used, empty vector
- A represents the linear coefficients for the inequality constraint – described in Equation 7
- *Aeq* represent the linear coefficients for the equality constraints – not used, empty matrix
- *lb*, *ub* represent the lower and upper bounds of the decision variable – lower bounds set to edge nodes, upper bounds set to all nodes in the network

$$min f^T x subject \begin{cases} x(intcon) & \text{are integers} \\ A.x \leq b \\ Aeq.x = beq \\ lb \leq x \leq ub \end{cases} \tag{6}$$

$$-2.w_{n_a} - w_{n_b} \leq -1 \tag{7}$$

3.2 Bandwidth Utilization and Packet Loss Calculation

With the *critical nodes* identified we can apply the network monitoring on the most important parts of the network. The results of NPPs monitoring will be combined with identified influence among the network elements. For the calculation we create set of equations and inequalities based on the following assumptions:

- packet loss on link can be expected if the link's load has reached the value of link's bandwidth
- the amount of incoming traffic is equal to the amount of outgoing traffic and packet loss for every node
- the traffic flow entering the node via interface i must be forwarded out via another interface
- the amount of traffic flowing through a link cannot be greater than the links' bandwidth

The number of equations is dependent on the density of the network. Equations are created for each node separately and for each possible path combination through the node. The inequalities are created for each link in the network. In general, the set of equations and inequalities are depicted in Equation 8 and Equation 9 respectively.

$$\forall n_p \in N : \forall l_{ip} \in L : LD_{l_{ip}} = \textstyle\sum_{j=1}^{S} LD_{l_{pj}} + PL_{l_{pj}}; j \neq p \quad (8)$$

$$\forall l_{ij} \in L : LD_{l_{ij}} \leq BW_{l_{ij}} \quad (9)$$

4. Results

Our proposed solution was verified in a simulation environment of Matlab R2018b with the Optimization toolbox installed and in a virtual machine with Ubuntu operating system. We created five testing topologies and for each one we created three versions with different characteristics to represent sparse, medium and dense networks. For every combination we decided to calculate with four different setups regarding the number of *edge nodes*. With this approach we created 60 different network setups on which our proposed solution was tested as described in Table 1.

The verification of our proposed solution was divided into three sections:

- selection of critical nodes
- calculation of bandwidth usage and packet loss
- application of acquired results into SDN scenario

The validation results for selection of *critical nodes* are depicted in Figure 2 – Figure 4. The proposed model took into consideration all possible combinations of the *edge nodes*. We focused on the number of selected nodes in different networks and the influence of various network characteristics on the final number of *critical nodes*. The results are represented in percentual values where 100% defines all nodes in the network.

The average amount of needed nodes for effective monitoring is around 64% of all nodes in the network. This shows 36% reduction of needed network nodes for overall monitoring, not depending on the size of the network as described in Figure 2. The density of the network slightly influences the amount of needed network nodes as described in Figure 3. Figure 4 depicts the direct impact of number of edge nodes on the amount of needed network nodes.

Table 1: Parameters of Testing Networks

Topology	S	avg(d)	Number of edge nodes			
			2	S/4	S/3	S/2
1	8	2.3	2	2	3	4
		3.3				
		4				
2	10	2.4	2	3	3	5
		3.2				
		4.2				
3	12	2.3	2	3	4	6
		2.8				
		3.8				
4	20	2.3	2	5	7	10
		2.9				
		4.2				
5	30	2.3	2	8	10	15
		3.1				
		4				

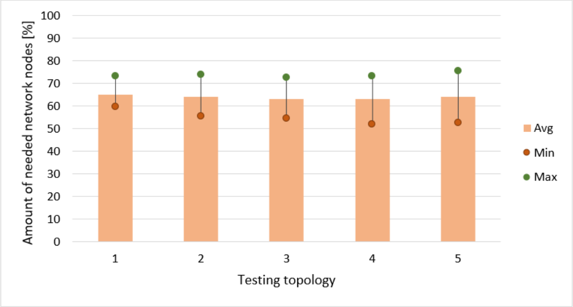

Figure 2: Nodes for monitoring per testing topology.

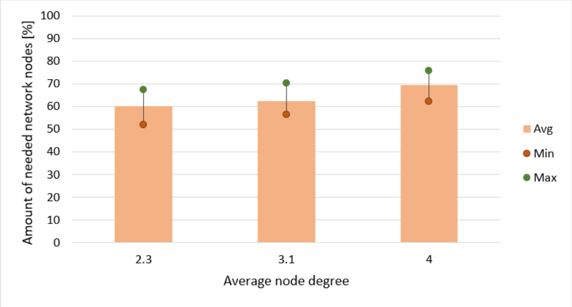

Figure 3: Nodes for monitoring per average node degree.

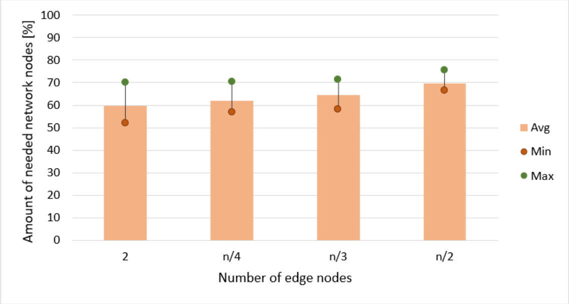

Figure 4: Nodes for monitoring per number of edge nodes.

The packet loss calculations were done on the same topologies as the testing of *critical nodes* selection. Ten scenarios of traffic flows for each of the networks were created and the bandwidth usage and potential packet loss was calculated. For each scenario different set of source-destination pairs for traffic flow was used and different amount of data was transferred, each of these generated randomly. The results of the testing scenarios for every testing topology are shown in Figure 5. ”Calculated loss” represents the exact value of packet loss calculated by our model based on information from monitored *critical nodes* and network topology. ”Estimated loss” represents an estimation of packet loss with lower and upper bounds. The average knowledge about the bandwidth usage and packet

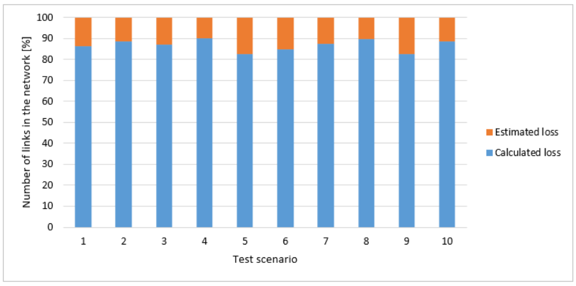

Figure 5: Packet loss calculation.

loss in the network was 86.7%. We didn't find any correlation with specific network type, network size or number of edge nodes in the network.

For every testing scenario used for selection of *critical nodes* we created an SDN topology with Floodlight controller using OpenFlow 1.3. We simulated monitoring of interfaces on each network node and compared the results with monitoring of selected *critical nodes*. Figure 6 shows the results per testing topology. It is obvious that the amount of artificial traffic is not dependent on the size of the network since the average amount of monitoring traffic is lowest in the biggest topology. The test results grouped by an average node degree of the network, depicted in Figure 7, show that the growing density of the network results in a higher percentage of needed artificial traffic. Figure 8 shows the results of testing grouped by the number of edge nodes in the network. As expected, the amount of artificial traffic needed for monitoring grows with the number of edge nodes in the network.

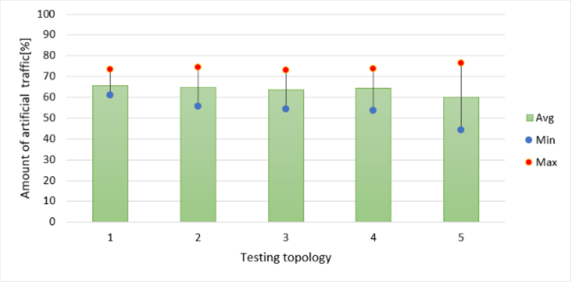

Figure 6: Artificial traffic per topology.

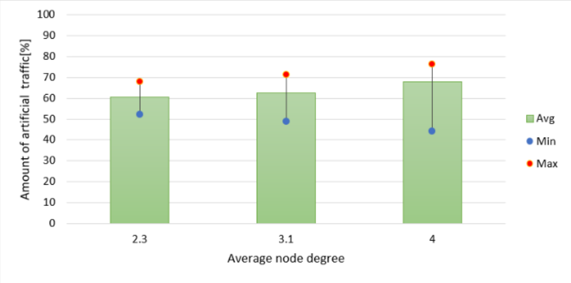

Figure 7: Artificial traffic per average node degree.

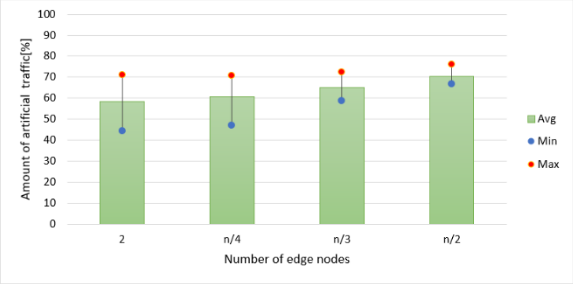

Figure 8: Artificial traffic per number of edge nodes.

5. Conclusions

The goal of our work was to design a novel network monitoring model to lower the monitoring overhead in the network while getting the overall knowledge of the network performance parameters. The proposed monitoring model analyzes the network, relationships among its elements and selects the most important elements. The selection is based on multi-objective optimization problem solving. The main objective was to cover all links in the network while positioning the monitoring probe at as few nodes as possible. The network monitoring may be then applied on much smaller number of nodes and links which will, in turn, decrease the amount of time and resources used.

The functionality of proposed model was verified in simulation environment of Matlab R2018b using 60 different network topologies. The results show that the number of nodes needed to be monitored ranges between 50-86.7% of all nodes in the network. The exact number is dependent of the density of the network and on the number of *edge nodes*. The possibility to calculate the remaining information about non-monitored elements was tested on ten different test scenarios for each of testing topologies. The average amount of calculated knowledge about the bandwidth usage and packet loss in the network was 86.7% while the remaining 13.3% were correctly estimated with lower and upper boundaries set. The amount of needed artificial monitoring traffic ranges from 42.6% up to 84.3%. The bandwidth consumption is therefore lowered by 15.7% - 57.4% which can be considered as a significant reduction of bandwidth consumption.

Acknowledgements. This research was supported by the Ministry of Education, Science, Research and Sport of the Slovak Republic, Incentives for Research and Development, Grant No.: 2018/14427:1-26C0. It was also partially supported by the Slovak Cultural and Educational Grant Agency (KEGA 011STU-4/2017) and the Slovak Science Grant Agency (VEGA 1/0676/12).

References

[1] Sihyung Lee, Kyriaki Levanti, and Hyong S. Kim. Network monitoring: Present and future. *Computer Networks*, 65, 2014.

[2] M. Hrubý. Optimization of network traffic flow. dissertation thesis. slovak university of technology in bratislava. 2013.

[3] Software-defined networking (sdn) definition. [online]. available at: https://www.opennetworking.org/sdn-definition/.

[4] W. Zhang, X. Zhang, H. Shi, and L. Zhou. An efficient latency monitoring scheme in software defined networks. *Future Generation Computer Systems*, 83, 2018.

[5] W. Queiroz, M. A. Capretz, and M. Dantas. An approach for sdn traffic monitoring based on big data techniques. *Journal of Network and Computer Applications*, 131, 2019.

[6] Z. Yu, Y. Zhang, Y. Zhu, D. Zhu, and P. Lin. Performance monitoring nodes deployment strategies for power wireless private networks based on improved mixed greedy algorithm. *IEEE International Conference on Energy Internet (ICEI)*, 2018.

[7] R. Umair, K. Shahid, and R. L. Olsen. Information reliability in smart grid scenario over imperfect communication networks using iec-61850 mms. *IEEE EUROCON 2017 - 17th International Conference on Smart Technologies*, 2017.

[8] Solarwinds network performance monitor. [online] available at: http://www.solarwinds.com/solutions/network-availability-monitor.aspx.

[9] Nagios. [online] available at: https://www.nagios.com.

[10] Total network monitor. [online] available at: http://www.softinventive.com/total-network-monitor.

[11] Zabbix: The enterprise-class open source network monitoring solution. [online] available at: https://www.zabbix.com.

[12] Paessler prtg network monitor. [online] available at: https://www.paessler.com/prtg.

[13] R. Khan and S. U. Khan. Design and implementation of an automated network monitoring and reporting back system. *Journal of Industrial Information Integration*, 9, 2018.

[14] M. Ljubojevic, A. Bajic, and D. Mijic. Centralized monitoring of computer networks using zenoss open source platform. *17th International Symposium INFOTEH-JAHORINA (INFOTEH)*, 2018.

[15] P. Zhang, H. Jin, Z. He, H. Leung, W. Song, and Y. Jiang. Igs-wbsrm: A time-aware web service qos monitoring approach in dynamic environments. *Information and Software Technology*, 96, 2018.

[16] A. Villegas, P. Perez, J. J. Ruiz, and J. Lopez-Poncela. Scalable monitoring of end-to-end delay in live video services. *IEEE International Conference on Consumer Electronics (ICCE)*, 2018.

[17] V. Demianiuk, S. Gorinsky, S. Nikolenko, and K. Kogan. Robust distributed monitoring of traffic flows. *IEEE 27th International Conference on Network Protocols (ICNP)*, 2019.

[18] K. Watabe, S. Hirakawa, and K. Nakagawa. Accurate delay measurement for parallel monitoring of probe flows. *13th International Conference on Network and Service Management (CNSM)*, 2017.

[19] K. Watabe, N. Murai, S. Hirakawa, and K. Nakagawa. Accurate measurement technique of packet loss rate in parallel flow monitoring. *28th International Conference on Computer Communication and Networks (ICCCN)*, 2019.

[20] P. Megyesi, A. Botta, G. Aceto, A. Pescapé, and S. Molnár. Challenges and solution for measuring available bandwidth in software defined networks. *Computer Communications*, 99, 2017.

[21] X. Zhang, Y. Wang, J. Zhang, L. Wang, and Y. Zhao. A two-way link loss measurement approach for software-defined networks. *IEEE/ACM 25th International Symposium on Quality of Service (IWQoS)*, 2017.

[22] J. Suárez-Varela and P. Barlet-Ros. Flow monitoring in software-defined networks: Finding the accuracy/performance tradeoffs. *Computer Networks*, 135, 2018.

[23] Z. Yang and K. L. Yeung. Flow monitoring scheme design in sdn. *Computer Networks*, 167, 2020.

[24] G. Tangari, D. Tuncer, M. Charalambides, Y. Qi, and G. Pavlou. Self-adaptive decentralized monitoring in software-defined networks. *IEEE Transactions on Network and Service Management*, 15(4), 2018.

[25] H. Tahaei, R. B. Salleh, M. F. A. Razak, K. Ko, and N. B. Anuar. Cost effective network flow measurement for software defined networks: A distributed controller scenario. *IEEE Access*, 6, 2018.

[26] intlinprog, mixed-integer linear programming (milp). [online] available at: https://www.mathworks.com/help/optim/ug/intlinprog.html.

Selected Papers by the Author

I. Hucková, P. Čičák. Advanced network monitoring using the selection of critical nodes. In *TSP 2019: 42nd International conference on telecommunications and signal processing*, pages 290–293, Budapest, Hungary. July 1–3, 2019. IEEE.

I. Hucková, M. Hrubý. QoS-Based optimization of data flow in MPLS networks. In *SAMI 2015. IEEE 13th international symposium on applied machine intelligence and informatics*, pages 83–88, Herľany, Slovakia. January 22–24, 2015. IEEE.

I. Hucková, P. Čičák. Advanced network monitoring and performance prediction. In *IDS 2015: 10th International Doctoral Seminar Proceedings*, pages 53–57, Varaždin, Croatia. September 24, 2015. Faculty of Organization and Informatics, University of Zagreb.