Performance comparison of QoS routing algorithms applicable to large-scale SDN networks

Slavica Tomovic, Igor Radusinovic
Faculty of Electrical Engineering
University of Montenegro, Montenegro
slavicat@ac.me, igorr@ac.me

Neeli Prasad
CTIF-USA (Aalborg university),
Princeton, New Jersey, USA
np@es.aau.dk

Abstract— **In Software Defined Networking (SDN) the network control plane is decoupled from the data plane and centralized at programmable controller. One of the advantages of centralized control is possibility to efficiently implement traffic engineering mechanisms, that are still lacking in today's backbone networks due to instability of distributed routing protocols and high management complexity. In this paper, we analysed suitability of different routing algorithms for dynamic setting up of performance guaranteed traffic tunnels in backbone SDN networks. Two routing constraints were considered: bandwidth and path delay. Beside bandwidth rejection ratio, which is commonly used as performance indicator for QoS (Quality of Service) routing algorithms, in the context of SDN low computational time is pointed out as important requirement.**

Keywords—QoS; routing; SDN; backbone.

I. INTRODUCTION

Today's networks are dominantly based on the Shortest Path First (SPF) routing algorithms that assign weighting factors to links statically. The route recalculation is triggered only when topology change occurs. Thus, network traffic is often unevenly distributed, and causes link congestion even when the total load is not particularly high. This is very problematic for multimedia applications that require certain QoS (Quality of Service) level for proper functioning.

The increasing demand for the multimedia applications has motivated a lot of research in the field of QoS provisioning. These efforts resulted in several proposed mechanisms, including IntServ [1] and DiffServ [2]. The main reason for their failure lies in too complex control plane that is distributed over all network devices [3]. Relatively recently, software defined networking (SDN) has been proposed to address this problem [4]. In SDN network, the control plane is decoupled from the data plane and moved to logically centralized controller. The potential of this concept reflects in the fact that traffic engineering (TE) techniques could be more efficiently and intelligently implemented in system having global view of the network state and applications' requirements. SDN implies programmability of network devices, so their behaviour could be dynamically handled by appropriate software to ensure optimal allocation of network resources and avoid congestion. For this reason, we believe that with SDN QoS-aware TE algorithms have strong potential for practical applications and wide adoption.

In this paper, QoS routing algorithms were analysed in terms of their suitability for establishing traffic tunnels in large-scale backbone SDN/OpenFlow networks [5]. Besides providing the required QoS level to service providers who rent resources of the backbone network, it is desirable that algorithm maximizes utilization of the network resources, since that is the main interest of the infrastructure provider. In that regard, bandwidth rejection ratio (BRR) has been defined as one of the metrics to evaluate algorithm's performance. QoS algorithms proposed in literature are dominantly focused on bandwidth guarantees [6-11]. We compared the solutions relevant for the analysed scenario. The average route length has been considered as another performance metric, because in WANs (Wide Area Networks) longer paths usually entail higher delay [12]. We showed that BRR and route length metrics are in conflict, and algorithms that perform the best according to first criterion are unable to satisfy the second criterion even under low traffic load. As high delay negatively reflects to providers of interactive multimedia services, performance of delay-bandwidth constrained algorithms was evaluated as well. While in the literature some of these algorithms were evaluated under assumption that each traffic flow is delay-sensitive [13], we compared them in more realistic scenario where flows may request only bandwidth guarantees. SDN controller scalability is highlighted as a challenge, so we provided illustrative results that show how techniques for reducing computation time of bandwidth-delay constrained algorithms degrade performance of the same.

The rest of the paper is organized as follows. In Section II traditional QoS architectures are briefly discussed and the considered system model is presented. In Section III the routing algorithms for bandwidth guarantees are described and compared. Bandwidth-delay constrained algorithms are discussed and evaluated in Sections IV. Conclusion remarks are given in Section V.

II. THE SYSTEM MODEL

Two main QoS architectures available today are: IntServ and DiffServ. However, neither of them is globally implemented due to inherent weaknesses. IntServ allows end-to-end, connection-level QoS guarantees through techniques of resource reservation. QoS requests are sent over a shortest path determined by traditional routing protocols. If that path is

---

This work is supported by the Ministry of Science of Montenegro under grant 01-451/2012 (FIRMONT) and EU FP7 project Fore-Mont (Grant Agreement No. 315970 FP7-REGPOT-CT-2013) http://www.foremont.ac.me.

978-1-4799-8569-2/15/$31.00 ©2015 IEEE

congested, the request would be rejected even though some other path to the same destination has sufficient bandwidth available. DiffServ architecture uses a different approach. Instead of resource reservation for each traffic flow, DiffServ marks packets at the network entrance and classifies them in finite number of traffic classes. Although this solution is more scalable then IntServ, it provides only relative performance guarantees. DiffServ also does not include new routing mechanisms. Multi Protocol Label Switching [14] (MPLS) provides a partial solution with its TE capability. However, TE mechanisms are not implemented in the networks of today's service providers due to inflexibility of the underlying protocols which do not allow real-time network reconfiguration [15].

We consider model of SDN/OpenFlow backbone network illustrated in Fig. 1, where the network intelligence is shifted from network devices to programmable controller. Implementation details about our QoS-aware OpenFlow controller can be found in [16]. The key functional blocks the design involves are: resource monitoring, route calculation, resource reservation and admission control. The controller has global view of network topology and uses OpenFlow to obtain information about traffic flows, TE-tunnels, ports, buffers, their state and statistics. Service providers negotiate QoS with the SDN controller, which calculates routes and reserves resources for the traffic flows. The controller also maintains information about each established QoS tunnel, including the path and bandwidth that had been assigned. When OpenFlow switch receives a packet, it tries to classify it in one of the existing flows. If no match is found, packet will be sent to the controller, which decides how corresponding flow should be handled. After decision making, new entry is added to device’s flow table. Beside flow definition, the entry contains a list of actions and statistical counters. Actions allow manipulating with headers, to forward packets to output ports or specific queues on them, to drop packets or encapsulate and send them to the controller. Traffic flow can be defined very flexibly, using any subset L2-L4 fields supported by OpenFlow specification [5], along with the identifier of the interface on which packet had arrived.

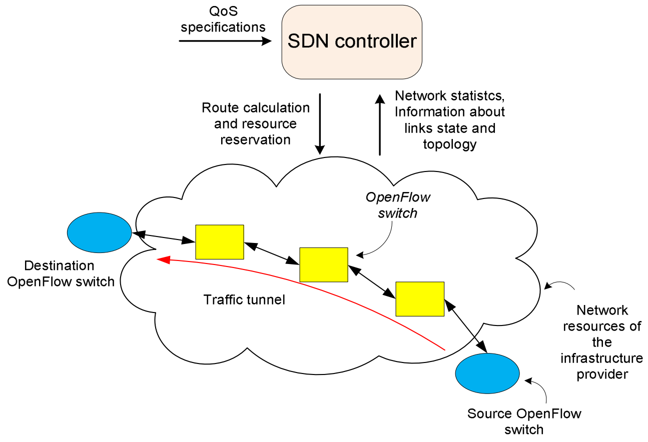


Fig. 1. SDN/OpenFlow model of backbone network.

The QoS algorithms analysed in the rest of the paper could be implemented within route calculation module of the proposed controller design. It should be noted that in [16] scenario with single SDN controller was assumed, which may not be acceptable for WANs due to scalability issues. This could be resolved by adding interface for inter-controller communication. However, potential interaction between multiple controllers (all of which require global network view) is very critical [17]. Therefore, the largest possible degree of physical centralization should be an important objective.

III. PERFORMANCE COMPARISON OF BANDWIDTH-CONSTRAINED ALGORITHMS

The most widely used bandwidth-constrained routing algorithm is MHA (Minimum Hop Algorithm) [9]. In MHA the shortest path between the source and the destination node is chosen. The algorithm maintains information about available bandwidth on each of the links, and only those that have enough resources to satisfy user's requirement are taken into account for routing. Although MHA is very simple, it could quickly create a bottleneck for future requests, leading to poor utilization of network resources. In [10] modified version of MHA was proposed, known as WSP (Widest Shortest Path) algorithm, which from the set of shortest feasible paths chooses the one with the highest amount of bandwidth available - the widest path. In this way the algorithm tries to make trade off between two conflicting requirements: load balancing and resource consumption. The opposite kind of compromise was made by SWP (Shortest Widest Path) algorithm [11], which computes the widest feasible path. If there is a more than one, path with the smallest number of hops is considered optimal.

The previously described algorithms use information about network topology and available bandwidth, but do not use information about source-destination (SD) node pairs to find a feasible route. Consequently, the selected route often constitutes an obstacle for a future demands, that is "interfere” with potential future path setups between the other SD pairs. Thus, more advanced routing algorithms were proposed with ambition to minimize "interference" on routes that may be critical to satisfy a future demands. Among these algorithms ([6-8]) we analysed performance of MIRA (Minimum Interference Routing Algorithm) [6] and DORA (Dynamic Online Routing Algorithm) [7], that do not assume *a priori* knowledge of traffic scheme. MIRA introduced concept of link criticality to determine which links to avoid in selecting a route. These critical links are identified by the algorithm as links with the property that whenever their capacity is reduced by 1 bandwidth-unit, the maxflow value of one or more the other SD pairs also reduces by 1 bandwidth-unit. This maxflow value is an upper bound on the total amount of bandwidth that can be routed between that SD pair [6]. Level of link criticality is proportional to number of pairs it interferes with. Hence the goal of MIRA is to find the least critical feasible path. Similarly, DORA algorithm tries to avoid routing over links with low residual bandwidth and with high potential to be included in paths of the future demands.

DORA operates in offline and online phases. In an offline phase array of path potential values (PPV) is calculated for each SD pair. Elements of PPV array correspond to network links and reflect their importance for the other SD pairs. Initially all PPV values are set to zero. Then, for corresponding SD pair the set of shortest disjoint paths is calculated. PPV values of links included in these paths are reduced by 1. Finally, each link is checked for occurrence in the sets of disjoint paths of the other SD pairs. If found there, its PPV value is increased by 1. The PPV values are determined for each SD pair separately, but these calculations are done only when network is initialized or when some change in topology happens. In an online phase, the PPV and residual bandwidth of each link are combined together to form the link weight. The impact of residual bandwidth is controlled by BWP (BandWidth Proportion) parameter:

$$weight = (1 - BWP) \cdot PPV + BWP \cdot \frac{1}{residual\_bandwidth}, \ 0 \le BWP \le 1 \quad (1)$$

Dijkstra algorithm ([18]) is used to compute a weight-optimized feasible path for QoS request.

To compare performance of bandwidth-constrained algorithms (and bandwidth-delay constrained algorithms in Section IV) we made special Python scripts. This allowed us to easier integrate these algorithms with our QoS OpenFlow controller which is also Python based [16], and confirm results in Mininet [19] later. In simulations we used MIRA topology, adopted from the literature dealing with the correlated routing problem [6-8][13]. The topology is illustrated in Fig. 2. The thicker links have a capacity of 4800 bandwidth units while the thinner links have a capacity of 1200 bandwidth units. The figure also shows the location of four different source–destination pairs, denoted as (S1, D1), (S2, D2), (S3, D3) and (S4, D4). A total of 500 static path setup requests were generated. The requests were uniformly distributed among the SD pairs, with the bandwidth requirement randomly chosen between 10, 20, 30 and 40 bandwidth units. The algorithms processed requests one by one, without *a priori* knowledge regarding future demands.

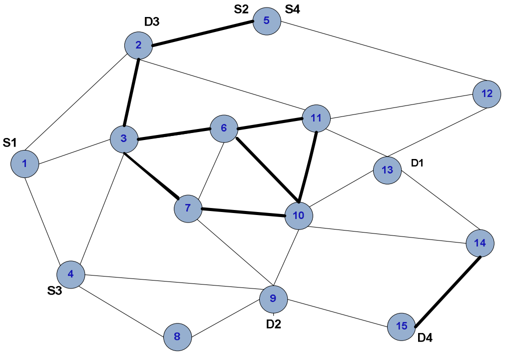


Fig. 2. MIRA topology

The two main performance metrics used to evaluate the algorithms are the bandwidth rejection ratio (BRR) and the average route length. The first metric has been introduced as replacement for path-setup rejection ratio, that is commonly used for performance comparison of QoS algorithms. In our simulations QoS request have different bandwidth requirements, so a low path-setup rejection ratio does not necessary reflect high efficiency. Hence, BRR metric is more suitable, since it is computed as the ratio of the amount of blocked traffic and the amount of bandwidth requirement of all generated requests.

In Fig. 3 the BRR for different algorithms is presented as a function of number of generated QoS requests. As expected, MHA algorithm started to reject requests before the other algorithms. This is consequence of using always the shortest feasible path for tunnel setup. The rapid bandwidth consumption of links along the shortest paths resulted very soon in the situation that no route between some SD pairs is feasible. On the other side, SWP algorithm, which aims to balance network load, performs well under low traffic load. However, when the load is heavy SWP performs poorly since it tends to allocate very long paths and consume a lot of resources. Contrary to SWP, the main performance objective of WSP is to reduce resource consumption by favouring shorter paths. Since resource preservation is very important when the network is congested, WSP performance is improved with the number of generated requests.

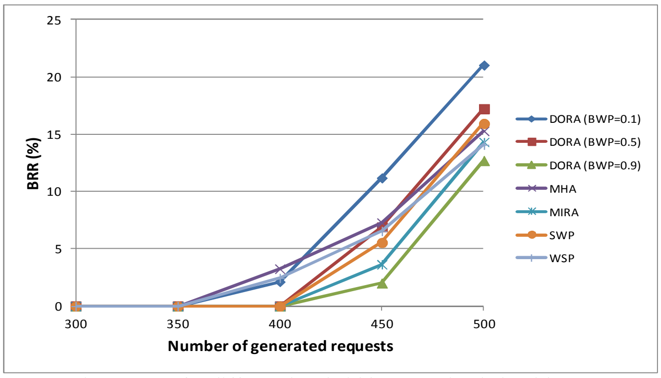


Fig. 3. BRR for different bandwidth-constrained algorithms.

Algorithms that consider "interference" between SD pairs proved to be the most efficient in terms of BRR. An exception is DORA algorithm when BWP parameter is set to low value (0.1). In that case DORA is focused on avoiding critical links, i.e. links with high PPV. The importance of link residual bandwidth is underestimated. MIRA algorithm performs much better, although it is based on similar idea. This is because MIRA defines interference more precisely, i.e. link is considered critical only if routing of 1 bandwidth unit over it reduces the maxflow value of one or more the other SD pairs. In order to run faster, DORA approximates this concept and it happens that some links are declared more critical than they actually are. Since critical links are determined offline, links that are not deficient with bandwidth could also be identified as critical. Thus, the link weight is not that good indicator of link importance for future QoS requests. With large values of BWP parameter DORA always gives the best results. That suggests the importance of dynamic routing and using residual bandwidth as a routing metric. In such configuration DORA estimates link criticality based on location of SD pairs, but always protects more those with low residual bandwidth.

Fig. 4 shows the average route length (in hops) for accepted connections as a function of number of generated requests. Route length was chosen as performance metric because it could indicate traffic delay. We did not use path delay as a metric because these algorithms do not consider delay as routing constraint. Therefore, random selections of link delay values would lead to unreliable results. On the other side, paths with larger number of hops introduce additional propagation delay if assume the same link attributes. As expected, among the TE algorithms, MHA and WSP perform the best. The load balancing solutions choose a longer paths if they have more bandwidth available. It is important to notice that DORA algorithm (BWP=0.9), which proved to be the best in terms of BRR, according to this criterion is almost always the worst. DORA's routes are often longer than the widest ones, selected by SWP algorithm. When BWP is set to 0.9, link weight is dominantly defined by reciprocal value of residual bandwidth, which is some kind of trade-off between minimum hop routing and load balancing. However, beside congested links, DORA avoids links with high PPV value. Hence, the routes are very long even under low traffic load.

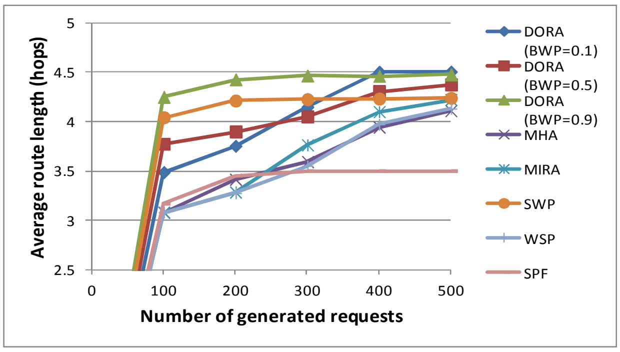


Fig. 4. Average route length for different bandwidth-constrained algorithms

The obtained results suggest that algorithms suitable for infrastructure providers (due to low BRR) could be unacceptable for delay-sensitive (DS) services. This type of services needs bandwidth-delay constrained algorithms.

IV. PERFORMANCE COMPARISON OF BANDWIDTH-DELAY CONSTRAINED ALGORITHMS

With the wide emergence of real-time Internet applications, delay-guaranteed algorithms have become increasingly required. Assuming that link delays are known, finding routes with delay guarantees is feasible with SPF algorithms. On the other hand, setting up QoS tunnels in backbone network includes resource reservation for the aggregated data flows. The problem of finding a route that satisfies bandwidth and delay requirements is still solvable in polynomial time [20]. Firstly, all links that do not have sufficient resources are pruned from the network graph. Then, SPF algorithm is used to compute the shortest path in terms of delay. If the path delay is less or equal than required, it is possible to provide required QoS. This algorithm was originally proposed in [20], and here we will denote it as (Minimum Delay Algorithm).

Besides providing the required QoS, it is important that the algorithm running on SDN controller efficiently uses the network resources in order to accept larger number of QoS requests. MDA algorithm was not designed with this in mind, and could not be considered as TE technique. The previous section presented three key approaches to traffic engineering: resource preservation, load balancing and "interference" avoidance. MDA does not follow strictly any of them. Among TE algorithms for bandwidth and delay guarantees, Maximum Delay-Weighted Capacity Routing Algorithm (MDWCRA) stands out [13]. MDWCRA is also based on idea of minimizing interference between SD pairs. However, it interprets concept of interference in a different way. The traffic tunnel is considered to cause interference if it is routed over bottleneck links of the shortest disjoint paths of the other SD pairs. Unlike DORA, MDWCRA calculates the shortest paths in terms of delay for each SD pair. The bottleneck links of these paths are defined as critical (C_{sd}), and MDWCRA avoids unnecessarily loading them. This is achieved by using weight function that increases with the level of link criticality:

$$w(link) = \sum_{(s,d):\, link \in C_{sd}} \alpha_{sd} \qquad (2)$$

In [13] three different weight functions were proposed. The first one is equal to the number of SD pairs for which the link is critical ($\alpha_{sd} = 1$). The second is inversely proportional to the total delay of the critical paths ($\alpha_{sd} = 1/delay_{s,d}$). The third is inversely proportional to the product of delay and bandwidth of the critical paths. Once the weights are determined, links with insufficient bandwidth are pruned from network graph, and Extended Dijkstra Shortest Path (EDSP) [21] could be used to compute the path with the lowest weight that satisfies delay requirement.

We examined performance of different bandwidth-delay constrained algorithms on MIRA topology (Fig. 2). In simulations fixed link delays were configured, uniformly distributed in range from 1ms to 50ms. It was assumed that propagation delay predominantly determines the total path delay. Scenario of WAN backbone network justifies this assumption, since in high-speed networks with aggregated traffic flows low queuing delays are expected. Furthermore, studies have shown that in real backbone networks effect of statistical multiplexing makes queuing delay negligible when load is below 90% [12]. In simulations the four SD pairs generated QoS requests for 10, 20, 30 or 40 bandwidth units with the equal probability. Beside MDWCRA and MDA algorithms, we evaluated performance of the algorithms presented in Section III, which had been extended with EDSP in order to ensure delay-constrained path selection. More precisely, once the algorithm determines link weights and eliminates links with insufficient bandwidth from the network graph, EDSP is used instead of ordinary Dijkstra to find weight-optimized path that satisfies delay requirement. In [13], MDWCRA was compared with some of these algorithms under the assumption that whole traffic has strict delay requirements. Here we extended the list of analysed algorithms and performance metrics, and considered more realistic scenario where not all traffic is delay-sensitive (e.g. FTP, Email, Web traffic). Percentage of DS requests was set to 70%, while their delay requirements varied in the range from 95 to 115ms.

The simulations results in terms of BRR are presented in Fig. 5. The prefix "DS" is used to denote modified, delay-sensitive versions of the algorithms from the Section III. The worst results were obtained for MDA algorithm that always selects path with the least delay. Although this is optimal from the perspective of DS applications, it happens that links critical for many SD pairs become quickly congested. These are links with the very low delay, which are usually part of the least delay paths. MDA unnecessarily routes non-delay sensitive (NDS) tunnels over them, and thereby reduces chances for DS requests to be accepted. DS MHA proved to be the second worst in most simulations. As links on the shortest paths were getting overloaded, number of feasible paths for DS traffic was reducing. DS WSP is based on a similar idea, and depending on workload and distribution of link delays its performance was slightly better or worse than that of DS MHA. BRR of DS MIRA algorithm is significant, although it uses information about SD pairs to make routing decisions. This is because DS MIRA defines critical links only based on residual bandwidth between the pairs, without considering importance of the link for future DS requests. From Fig. 5(c) it seems that DS MIRA performs well for NDS traffic, but this is mostly consequence of blocking large number of DS requests, due to which more resources stays free. DS SWP performs good when the network load is low. By forcing paths with the most bandwidth DS SWP efficiently balances load over the links, which resulted in small BRR for smaller numbers of generated request. With a load increase the performance degrades, mostly because NDS traffic is routed over very long paths, consuming a lot of resources. Performance of DS DORA depends on value of BWP parameter. The best results were obtained when BWP was set to 0.9. By balancing network load DS DORA alleviates deficiencies of DS MIRA. Moreover, it slightly outperforms MDWCRA is some situations. The results shown in Fig. 5 refer to the case when the third definition of weight function was used for MDWCRA. The two other weight functions resulted in slightly worse performance. Although it seems that MDWCRA's logic for selecting critical links is more appropriate for the scenario with DS traffic than the logic of DS DORA algorithm, that is not always the case. Depending on network topology it could happen that some of the disjoint paths with the least delay are completely useless for DS traffic, but are still avoided for routing. Also, MDWCRA treats all non-critical links in the same way, which causes more unbalanced resource consumption.

Fig. 6 shows that the modified version of DORA algorithm is the most efficient in load balancing. Standard deviation of residual bandwidth on links is important indicator of the algorithm's ability to distribute traffic tunnels in the network. The smaller this parameter is, the network is more resistant to link failures because there is higher probability for finding alternative routes.

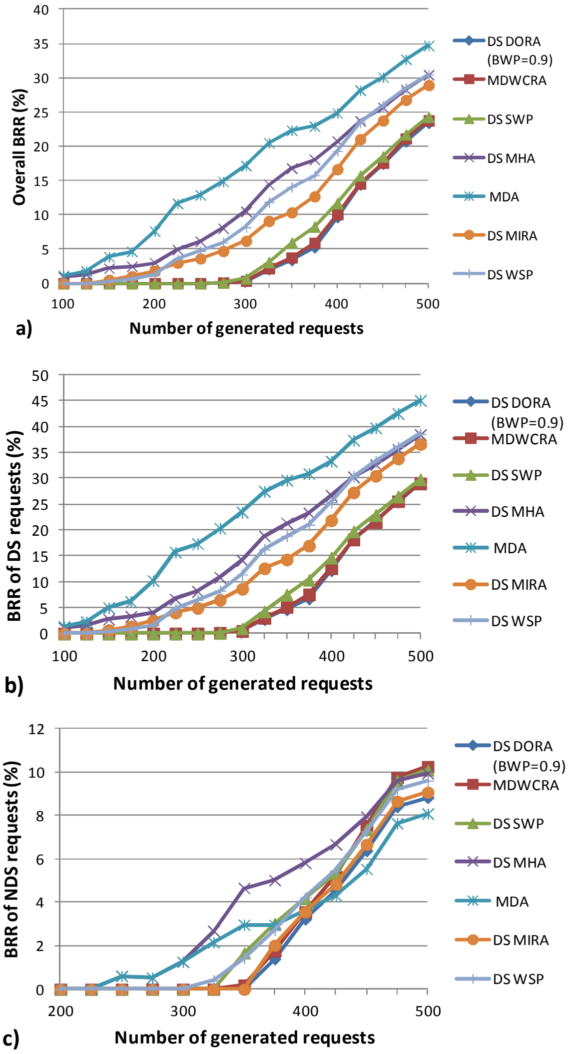


Fig. 5. Comparison of different bandwidth-delay constrained algorithms in terms of BRR performance for all (a), DS (b) and NDS (c) requests.

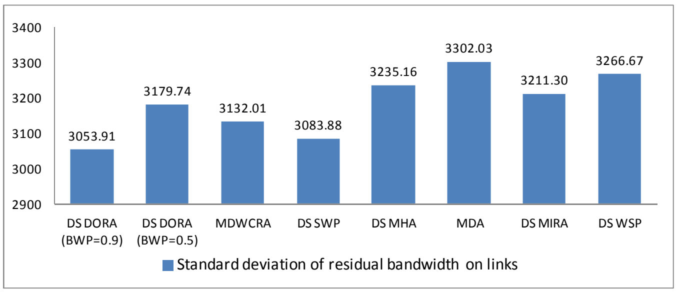
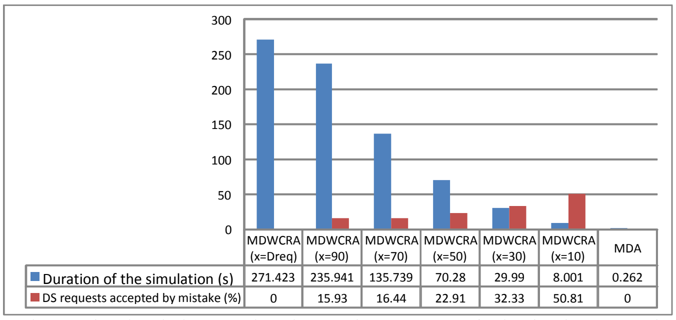


Fig. 6. Standard deviation of residual bandwidth on links.

Although MDWCRA and DS DORA provide significantly better BRR results than MDA, this comes at cost of very high complexity. Like MDA, these algorithms consider two routing constraints: bandwidth and delay, but additionally path weight/cost is trying to be optimized. This also applies to all the other analysed algorithms, except MDA. After eliminating links with insufficient bandwidth, the routing problem could be defined as constrained path optimization, which is known to be *NP-complete*. The problem could be simplified by discretizing values of the links' delays [21]. This is achieved by assigning a new weighting factor to each link:

$$w_2(link)=\left[\frac{link\_delay\cdot x}{delay\_requirement}\right] \qquad (3)$$

where x is a positive integer. In this way, delay constraint transforms in constraint: $w_2(path)\le x$. Efficiency of this technique depends on error introduced with the discretization (which increases with the network size). With a larger x, the error accounts for a smaller portion of the link delay. However, the computation overhead is directly related to x. To illustrate this problem, in the second set of simulations we increased number of generated requests to 1000 (60% was delay sensitive) and measured simulation running time and percentage of DS requests accepted by mistake. In Fig. 7 MDWCRA is compared with MDA in this regard for different values of x. It can be observed that MDWCRA processed requests much slower in all analysed situations. Also, small values of x considerably compromised the accuracy of the algorithm. For x=90 percentage of mistakenly accepted DS request was 15.93%, and for the value of x=10 even 50.81%. On the other side, if *w2* is rounded to the nearest integer, the higher BRR could be expected.



	MDWCRA (x=Dreq)	MDWCRA (x=90)	MDWCRA (x=70)	MDWCRA (x=50)	MDWCRA (x=30)	MDWCRA (x=10)	MDA
🟦 Duration of the simulation (s)	271.423	235.941	135.739	70.28	29.99	8.001	0.262
🟥 DS requests accepted by mistake (%)	0	15.93	16.44	22.91	32.33	50.81	0

Fig. 7. The simulation running time and percentage of mistakenly accepted DS requests for MDWCRA and MDA. *Dreq* denotes value of delay request.

V. CONCLUSION

SDN is an emerging networking paradigm that eliminates the need for distributed routing protocols. With centralized decision-making, it provides more flexibility in control and enables implementation of sophisticated routing mechanisms. In this paper we analysed performance of bandwidth constrained and bandwidth-delay constrained centralized routing algorithms in terms of their suitability for providing QoS in large-scale SDN backbone networks. Various metrics were used to evaluate their performance in more detail. It has been shown that bandwidth-constrained algorithms that use information about SD pairs can significantly increase network capacity. However, these algorithms choose very long routes regardless of network load, which may not be acceptable for providers of real-time services. On the other side, bandwidth-delay constrained algorithms can provide the required performance, but TE implies high computational time. Considering centralized nature of the SDN control plane, this could be seen as significant limitation. In our future work we aim to define SDN control framework that provides the best trade-off between these two incompatible requirements. This implies QoS model where DS traffic is classified in limited number of classes, so that potential routes can be calculated in advance and stored at SDN controller. Also, in our experiments we noticed that the network topology and number of SD pairs affect BRR performance of the analysed algorithms. This will be subject of the further research as well.

REFERENCES

[1] *Integrated services in the Internet architecture: An overview*, IETF, RFC 1633, June 1994.

[2] *An architecture for differentiated services*, IETF, RFC 2475, Dec. 1998.

[3] S. Das, "A unified control architecture for packet and circuit network convergence". PhD Dissertation, Stanford Univeristy, Department of electrical engineering, June 2012.

[4] Open Networking Foundation, "Software defined networking: the new norm for networks," Web white paper, retrieved Apr. 2015.

[5] OpenFlow switch specification v1.0.0, Retrieved Apr. 2014, Available at: http://archive.openflow.org/documents/openflow-spec-v1.0.0.pdf.

[6] M. Kodialam, and T. V. Lakshman. "Minimum interference routing with applications to MPLS traffic engineering," *IEEE INFOCOM*, vol. 2., 2000, pp. 884 - 893.

[7] R. Boutaba, W. Szeto, and Y. Iraqi. "DORA: Efficient routing for MPLS traffic engineering," *Journal of Network and Systems Management10.3*, 2002, pp. 309-325.

[8] S. Suri, M. Waldvogel and P. R. Warkhede. "Profile-based routing: A new framework for MPLS traffic engineering," *Quality of future Internet Services*, Springer, 2001, pp. 138-157.

[9] Q. Ma, and P. Steenkiste. "On path selection for traffic with bandwidth guarantees," *IEEE international conference on network protocols*, 1997, pp. 191-202.

[10] *QoS routing mechanisms and OSPF extensions*, IETF, RFC 2676, 1999.

[11] Wang, Zheng, and Jon Crowcroft, "Routing algorithms for supporting resource reservation," *IEEE JSAC*, 1996.

[12] C. Fraleigh, F. Tobagi and C. Diot, "Provisioning IP backbone networks to support latency sensitive traffic," *INFOCOM 2003,* vol. 1, 2003, pp. 375-385.

[13] Y. Yang, J.K. Muppala, and S.T. Chanson, "Quality of service routing algorithms for bandwidth-delay constrained applications," *Ninth International Conference on Network Protocols*, pp. 62-70, Nov. 2001.

[14] *Multiprotocol label switching architecture*, IETF, RFC 3031, Jan. 2001.

[15] S. Das et. al., "Application-aware aggregation and traffic engineering in a converged packet-circuit network," *OFC/NFOEC*, Mar. 2011, pp.1-3.

[16] S. Tomovic, N. Prasad and I. Radusinovic,"SDN control framework for QoS provisioning," *TELFOR*, Nov. 2014, pp.111-114.

[17] S. Sezer et. al., "Are we ready for SDN? Implementation challenges for software-defined networks," *IEEE Communications Magazine*, vol.51, no.7, July 2013, pp. 36-43.

[18] E. W. Dijkstra, "A note on two problems in connection with graphs," Numerische Math. 1, 1959., pp. 269-271.

[19] Mininet emulator: http://mininet.org/

[20] Z. Wang, J. Crowcroft, "Quality of service routing for supporitng multimedia applications," *IEEE Journal on selected areas in Comm.*,vol. 14., no. 7, pp. 1228-1234, Sep. 1996.

[21] S. Chen, "Routing Support for providing guaranteed end-to-end Quality-of-Service," PhD thesis, Computer Science, University of Illinois, Urbana-Champaign, 1999.