Entropy-based QoS Routing for Software-Defined Edge Network

Chih-Ho Hsu, Yao Chiang, Hung-Yu Wei
Dept. of Electrical Engineering
National Taiwan University
Taipei, Taiwan
hywei@ntu.edu.tw

***Abstract*—Recently, edge computing has been proposed as a promising architecture to enhance network performance. Nonetheless, due to the distributed nature of edge nodes, it also creates complexities to manage scarce resources and numerous service flows. In addition, since many emerging applications have unique Quality of Service (QoS) requirements in terms of throughput, delay and packet loss rate, how to efficiently route these heterogeneous service flows in edge network to guarantees their QoS becomes a challenging issue. Fortunately, the emergence of Software Defined Network (SDN) can serve as an enabler to fulfill the potential of edge computing by flexibly deploying routing strategies in the network. In this paper, by considering the effect of link utilization toward QoS, we aim to maximize the QoS of a set of incoming service flows through routing in the software-defined edge network while satisfying the QoS requirement of ongoing flows. To efficiently optimize the formulated routing problem, we proposed a heuristic routing algorithm based on the concept of entropy so that it can achieve load balance while maintaining QoS. Extensive simulation results show that the proposed scheme outperforms the traditional routing method and existing SDN routing algorithm in terms of QoS, load balance and satisfaction ratio.**

***Index Terms*—SDN, Routing, QoS, Load Balancing, Entropy, Edge Computing**

I. Introduction

Recently, edge computing has been proposed as a promising architecture to meet the emerging need for innovative applications by providing storage and computing resource at network edges. By processing service requests generated by various edge devices locally, edge computing can potentially enhance the network performance in a more efficient manner.

However, edge computing also involves many complexities in its management and orchestration. Owing to the scarce resource at edge nodes, these services are usually provided by different edge servers in a distributed manner. Thus, delivering the requested services to the end users may involve bi-directional data transmission among multiple edge nodes. Furthermore, many emerging applications have unique Quality of Service (QoS) requirements in terms of throughput, delay, packet loss rate or combination of these factors. For instance, high-definition video streaming requires tremendous throughput but is less strict to the initial delay. Also, interactive game like virtual reality demand ultra-low delay to reach acceptable user experience. On the other hand, some low-power and resource-constrained IoT devices are especially sensitive to packet loss [1]. Thus, given the fact that networking resources at edge nodes are rather limited, how to efficiently route these heterogeneous data flows in edge networks to guarantees their QoS becomes a challenging issue.

Fortunately, the emergence of Software Defined Network (SDN) provides a potential solution to the above problem [2]. SDN is a novel network paradigm separating the control plane from the data plane of the network node. In SDN, a logically centralized controller is deployed to communicates with the distributed software-defined forwarding element (SDFE) obeying protocols such as OpenFlow. In this way, SDN can maintain global view of the network, including topology, ongoing traffic and link state information. In contrast to traditional networks where only best-effort service is provided due to the fierce contention between different service flows, SDN can optimize the network performance by flexibly deploying routing strategies in the SDFE based on the QoS demand of requests and current network condition.

In general, the research of routing algorithm for SDN currently focuses on two aspects, which aim to achieve load balancing of the network [3-9] and to satisfy heterogeneous QoS requirements of the requests [10-16], respectively. As for the former, their objective can be described as minimizing the maximum link utilization (LU) in the network. As noted in [3], [17-18], the delay and packet loss at the links are increasing functions of LU. As a result, the network can provide decent QoS for the whole service flows when their objective (i.e. maximum LU) is optimized. Nevertheless, even though routing for load balancing can reduce the QoS degradation to some extent, their routing result may not guarantee the QoS of the whole service flow. On the other hand, though the routing scheme for QoS guarantee can ensure the QoS of each individual service flow, they often consider the delay and packet loss rate (PLR) of links as a static constant, neglecting the effect of LU toward the QoS in the network. This may make their routing strategies less effective in real world scenarios when many of QoS flows are routed in few certain links.

Motivated by the above work as well as the challenges, in this paper, we aim to maximize the QoS of a set of flows while satisfying the QoS constraints of ongoing flows as well as the new flow in the network by combining the objective of load balancing and QoS optimization. Specifically, our main contributions are summarized as follows:

TABLE I
A COMPARISON OF RELATED WORK IN THE LITERATURE

#	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**We**
Optimized Metric	MLU	MLU	MLU	MLU	MLU	MLU	MLU	D	D, PLR	D, PLR, BW	D, PLR	BW	BW	Cost	QoS
Methodology	FPTAS	Heu	FPTAS	Lagrange	Heu	RL	Heu	AHP	Greedy	DQL	LARAC	Heu	B&C	Heu	Heu
Architecture	N/A	N/A	Cellular	N/A	Edge	N/A	N/A	Cloud	Edge	Cloud	N/A	N/A	Cloud	WMN	Edge
QoS awareness	X	X	O	X	X	X	X	O	O	O	O	O	O	O	O
Load balance	O	O	O	O	O	O	O	X	X	X	X	O	X	X	O
Routing scheme	MF	MF	MF	MF	MF	MF	MF	MF	MF	SF	SF	SF	SF	SF	MF

Hints: **MLU**=Maximum Link Utilization, **D**=Delay, **PLR**=Packet Loss Rate, **BW**=Bandwidth, **Heu**=Heuristic, **FPTAS**=FastPolynomial Time Approximation Schemes, **AHP**=Analytic Hierarchy Process, **LARAC**=Lagrangian Relaxation Based Aggregated Cost, **B&C**=Branch and Cut, **DQL**=Deep Q-Learning, **MF**=Multiple Flow, **SF**=Single Flow

1) By considering the QoS parameters as a function of link utilization, we design a more reasonable objective for QoS routing problem in software-defined edge network. In this way, we can determine precise routing decisions so that the network performance will be further enhanced.

2) We formulate the routing problem with heterogeneous QoS requirement as an integer non-linear programming, which is NP-hard. To efficiently optimize the problem, we proposed a low-complexity heuristic routing algorithm based on the concept of entropy. By designing two different optimization stages, it can reach decent QoS while balancing network loads.

3) We evaluate the proposed scheme using the RYU controller and the Mininet network emulator in real-world network topology. Simulation results show that the proposed scheme outperforms the classical OSPF, LARAC and greedy routing algorithm in terms of QoS, active links and satisfaction ratio.

The remainder of this paper is organized as follows. Sec. II discusses the related literature of our work. Also, our system model and problem formulation are organized in Sec. III. Then, the proposed method is presented in Sec. IV. Eventually, Sec. V elaborates the simulation results to evaluate our proposed method and Sec. VI closes this work with conclusions.

II. RELATED WORK

This section first briefly outlines existing efforts regarding the related topic of this paper, including routing for load balancing and routing for QoS guarantees. Afterward, we will present a comparison table to validate our contribution.

A. Load balancing

In [3-4,6], they explore the load balancing in the hybrid SDN network where only a portion of the forwarding elements can be controlled by SDNC and the rest of the network containing traditional switch using OSPF protocol. However, since existing load balancing schemes are mostly focused on core network only, which is inadequate for efficient end-to-end traffic delivery in mobile networks, a load balancing framework that exploits the topology and traffic information of both core and backhaul networks for SDN-based 5G networks is proposed in [5]. The author in [7] addresses the routing optimization problem in SDN-based IoT with Ternary Content Addressable Memory (TCAM) capacity constraint. Also, a Reinforcement Learning based scheme is proposed in [8], which can learn a policy to select critical flows for each given traffic matrix automatically. On the other hand, the aim of [9] is to jointly optimize the traffic matrix measurement and load balancing process under the TCAM capacity and flow aggregation constraints in software-defined networks. Finally, in [10], they propose a bi-objective optimization method to find an optimum path for packet transmission within a data center for enhancing cloud gaming.

B. QoS guarantees

In [11], they consider QoS routing strategies in software-defined IoT network and propose a greedy routing approach based on Yen's K-shortest paths algorithm to compute the optimal forwarding path, while considering the QoS requirements. However, they only consider 2 kinds of QoS requirements: either delay-sensitive or loss-sensitive. A novel routing strategy based on Deep Q-learning is proposed in [12] with a goal to achieve low latency/PLR for mice-flows and high throughput with low PLR for elephant-flows. Their model can learn from dynamic environment and generate routing paths autonomously for SDN-based data center networks without manual intervention. In [13], they solve optimization of dynamic QoS routing for scalable encoded video streaming as a constrained shortest path problem, where they classify the QoS flow into 2 levels: 1) the base layer for QoS guarantee and 2) the enhancement layers, which are treated as best-effort flows. The author of [14] proposes an effective heuristic solution to find multiple paths that can satisfy bandwidth requirements of a given set of unsplittable flows. In [15], they consider the QoS routing problem in wide area networks and resolve it in a polynomial time leveraging the Branch-and-Cut algorithm. Finally, a QoS routing algorithm for SDN-wireless mesh network is proposed in [16] to find out the path with minimum cost meeting the demands of QoS.

A comparison of related work in the literature is listed in TABLE I. We observe that so far there is not a paper simultaneously considering: 1) load balancing, 2) QoS awareness 3) multi flow routing in edge network. Therefore, we start to draw a design and implement these features.

III. NETWORK MODEL AND PROBLEM DESCRIPTION

In this paper, we study the routing problem in software-defined edge network for QoS guarantee. As shown in Fig. 1, our system compose of three main components: 1) SDN controller (SDNC), which is a logically centralized controller periodically recording services requests and link state information in the network, 2) a set of SDFE that are orchestrated

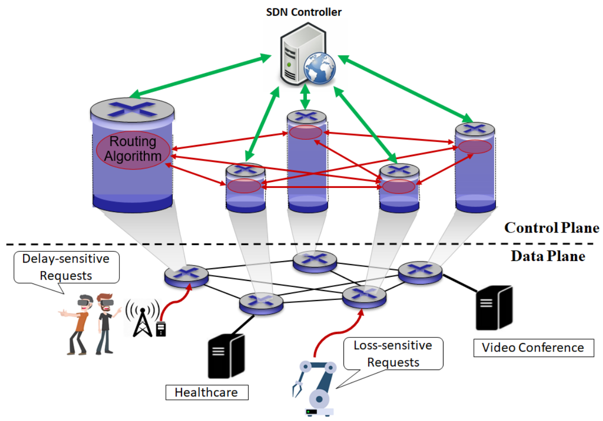

Fig. 1. Software-defined edge network with heterogeneous QoS requirement.

by SDNC to install forwarding rules and 3) a set of service requests (i.e. flows) with heterogeneous QoS requirement. In this section, we will focus on the mathematical description of the mentioned components and elaborate on the definition and constraints of our routing problem.

To begin with, we model the Software-defined edge network as a weighted undirected graph $G = \{V, E\}$, where $V = \{1, 2, 3, ..., n\}$ is the set of nodes, including edge server for providing different services and SDFEs controllable by SDNC, and $E = \{1, 2, 3, ..., m\}$ is the set of transmission links among edge nodes and each link is equipped with capacity $C(e), e \in E$. Also, considering that there exists a few service flow in the network, we denote $\theta(e)$ as the utilization of link e, which can be calculated by the throughput of ongoing flow in e divided by capacity $C(e)$. Then, in each system time, the SDNC will collect a set of services requests F (i.e. incoming QoS flows) with the specific requirements from edge nodes. Mathematically, F is given as:

$$F = \{f_i | f_i = (s_i, d_i, T_i^*, L_i^*, R_i^*)\} \tag{1}$$

where f_i denote the i^{th} incoming flow, s_i, d_i indicte the source and destination of the flow, and T_i^*, L_i^*, R_i^* represent the requirement of delay, packet loss rate (PLR) and throughput of the flow f_i, respectively.

Each flow in the set F is either dropped if the network condition fails to provide the required QoS performance, or will be routed by SDNC through a single path. For the i^{th} flow, the set of feasible paths from source s_i to destination d_i that can satisfy its requirement can be described as:

$$P_i = \{p_{i,j} | p_{i,j} = (e_{i,j,1}, e_{i,j,2}, ..., e_{i,j,n_{i,j}})\} \tag{2}$$

where $p_{i,j}$ is the j^{th} possible routing path for the flow f_i, $e_{i,j,k}$ denote the k^{th} link that the j^{th} routing path will pass by and $n_{i,j}$ means the number of passing links in the path $p_{i,j}$. On the other hand, since rerouting an ongoing flow might negatively affect the QoS [8] (e.g. frequently changing the path of a TCP flow may cause packets loss, packet out-of-order and related problem), we only focus on guaranteeing QoS of incoming flows in this study.

As mentioned before, the goal of this paper is to maximize the QoS of overall flows in the network. To that end, we should first design the utility function of each flow. By assuming that Qos will be positively proportional to throughput and reduction of delay, PLR (i.g R, T, L), and each service flow has its own QoS preference depending on their service type, we define the utility funtion of flow f_i as:

$$\begin{aligned} U_i(p_i^*) &= \alpha_i \left(\frac{T_i^* - T(p_i^*)}{T_i^*} \right) + \beta_i \left(\frac{L_i^* - L(p_i^*)}{L_i^*} \right) \\ &+ \gamma_i \frac{R(p_i^*)}{R}, \qquad 0 \le \alpha_i, \beta_i, \gamma_i \le 1, \forall f_i \in F \end{aligned} \tag{3}$$

where $p_i^* \in P_i$ is the path decision of flow f_i, $\alpha_i, \beta_i, \gamma_i$ are preference coefficients of different factor and $T(p_i^*), L_i^*(p_i^*), R(p_i^*)$ is the experienced delay, PLR and throughput of the decided path p_i^* for the i^{th} flow. Note that in order to avoid the system in favor of a certain factor, we normanlize the each value into [0,1]. Also, R is a constant larger than $R(p_i^*)$ to keep the value less than 1.

However, as noted in [3], the delay and packet loss rate at each link is an increasing function of the link utilization θ. Instead of adopting the static model where the delay and PLR remain consistent in each link, we adopt the mathematical model formulated in [17-18] to describe the relation of θ toward delay (i.e. T) and PLR (i.e. L) when designing our objective function. As a result, by denoting $r_i(e), t_i(\theta(e)), l_i(\theta(e))$ as the throughput, delay, PLR of i^{th} flow experienced in link $e \in E$, we can obtain the following relation:

$$\begin{aligned} R_i &= \min\{r_i(e), \forall e \in p_i^*\} \\ T_i &= \sum_{e \in p_i^*} t_i\Big(\theta(e)\Big) \\ L_i &= 1 - \prod_{e \in p_i^*} \Big(1 - l_i\Big(\theta(e)\Big)\Big) \end{aligned} \tag{4}$$

Eventually, the problem is defined as to maximize QoS of a set of incoming flows while satisfying the QoS constraints of both incoming and ongoing flows by determining their routing path, which is formulated as follow:

$$\begin{aligned} &\max_{p^*} \sum_{f_i \in F} U_i(p_i^*) \\ \text{s.t.} \quad & C1: \sum_{f_i \in F \cup F'} r_e(i) \le C_e, \forall e \in E \\ & C2: \min\{r_i(e), \forall e \in p_i^*\} \ge R_i^*, \forall f_i \in F \cup F' \\ & C3: \sum_{e \in p_i^*} t_i\Big(\theta(e)\Big) \le T_i^*, \forall f_i \in F \cup F' \\ & C4: 1 - \prod_{e \in p_i^*} \Big(1 - l_i\big(\theta(e)\big)\Big) \le L_i^*, \forall f_i \in F \cup F' \end{aligned} \tag{5}$$

where F' is the set of ongoing flows and p^* is an integer vector denoting the set of routing decisions of whole flows, which imply that it is an integer non-linear programming problem, which is NP-hard. In our problem, the constraint $C1$ ensures that no link can be overloaded. The constraint $C2, C3$ and $C4$

Algorithm 1: Entropy-based balanced QoS routing scheme

Input: A set of incoming flow F and QoS demands
Output: The set of routing decision of whole flows p^*

foreach f_i *in* F **do**
  $P_i \leftarrow$ the set of feasible path with guaranteed QoS from s_i to d_i obtained by K-Shortest path alg.
  if $P_i \neq \phi$ **then**
    $p_i^* \leftarrow$ the path in P_i with maximum entropy increment, calculated by (8)
  else
    Drop the flow f_i
  end
end
while *not converge* **do**
  foreach f_i *in* F **do**
    $P_i \leftarrow$ the set of feasible path with guaranteed QoS, obtained by K-Shortest path alg.
    if $P_i \neq \phi$ **then**
      $p_i^* \leftarrow$ the path with best QoS based on p^*
    end
  end
end

guarantees that the experienced throughput, delay and PLR of each flow must satisfy its requirements, respectively.

IV. Entropy based Balanced QoS Routing

In this section, we propose a heuristic method named Entropy-based Balanced QoS Routing (EBQR) to efficiently optimize our formulated problem while providing better load balancing in the network.

To begin with, we first elaborate on the concept of entropy. Entropy is used to measure the uncertainty in information theory. Quantitatively, if the information has higher entropy, it is more stochastic. Also, the entropy will be always increased in any isolated system. Such characteristic has proven its effectiveness in solving problems of different areas [19] and may give some guideline to our system.

Adapted from the definition of entropy in infromation theory, presented as $H = \sum_i p_i \log(p_i)$ where p_i is the probability of an event i, we definte the total entropy of the routing system as:

$$H = -\sum_{e \in E} \theta(e) \log \theta(e) \tag{6}$$

To examine the feasibility of (6), we consider the case of $\theta = 0$. When θ approach to 0^+, the equation $\theta \log \theta = 0$ will hold, which can be simply shown with the L'Hôpital's rule:

$$\lim_{\theta \to 0^+} \theta \log \theta = \lim_{\theta \to 0^+} \frac{(\ln \theta)'}{(\frac{1}{\theta})'} = \lim_{\theta \to 0^+} -\theta = 0 \tag{7}$$

Afterward, by using the Lagrange multiplier, we can find out when $\theta(1) = \theta(2) ... = \theta(m) = \frac{1}{m}$, where m is the number of links in the network, the entropy H will reach the maximum. We can imply that under the same load, the whole system tends to be balanced and the load tends to be equally distributed when the value of H goes high. Thus, we adopt such concept in our heuristic routing algorithm.

We first analyze the difficulty of the problem before starting to present our algorithm. The difficulty of the problem (5) lies in the fierce contention for limited networking resources between multiple flows and it will be exhausting to search for optimal solution where the QoS of overall flows is maximized. Intuitively, we can simply route the flow one by one from the flow set to reduce the complexity. However, the final routing decisions in such scheme may not be optimal and will be affected by the ordering of flow to route (e.g. if many flows has occupied a cetain link before a target flow get routed, its best choic for routing may be different). Motivated by these challenges, the designed algorithm consists of two stages, presenting 2 different philosophies. In the first stage, the algorithm intends to maximize the entropy of the system so that the load tends to be balanced, which is not necessarily related to the objective of (5). Then, in the second stage, the algorithm will iteratively tune the transitive routing decision of each flow toward the solution that maximizes the objective of (5). In this way, we can explore more feasible solution in the solution space compared with the greedy method.

Specifically, to approach maximum system entropy in first stage, we defined the entropy increment (e.g. the difference between current and previous entropy) of routing decision on flow f_i (i.e. p_i^*) as:

$$\Delta H = H' - H = -\sum_{e \in p_i^*} \theta'(e) \log \theta'(e) + \sum_{e \in p_i^*} \theta(e) \log \theta(e) \tag{8}$$

where H', $\theta'(e)$ denote the system entropy and LU of e after routing the flow f_i on the path p_i^*.

As said in the maximum entropy principle, for an unknown distribution, the most reasonable implication is the most stochastic one fitting the given constraints [19]. Thus, since each flow has no way to know the routing decision of the rest of the flows, it is reasonable to select the path with maximum entropy increment so as to approach maximum entropy.

Based on the above explanation, in the first stage, the algorithm will determine the transitive routing decision of each flow in random order through selecting the path with maximum entropy increment from the set of the feasible paths, which can be obtained by Yen's K-shortest paths algorithm [20]. If there is no feasible path that can fulfill the QoS requirement of a flow, it will be dropped by SDNC. After all of the flow have selected their routing path, the second stage of the algorithm will continue to tune the obtained transitive routing decision. Different from the first stage, the algorithm will select the path with the best QoS from the set of feasible paths for each flow. This process will be repeated until it converges, which will happen when no flow can get better QoS by changing its current routing decision (e.g. the selected path is already the best path in the feasible set) or the defined maximum execution time is reached. Furthermore, to avoid the oscillation in routing decisions, the algorithm would record the historical value of p^* and will be terminated if oscillation occur. The whole process is presented in Alg. 1. Note that p_i^* indicates the selected path of flow f_i and p^* denote the

TABLE II
SIMULATION PARAMETERS

Parameters	Value
Link capacity	$[10^2, 10^3]$Mbps
Delay requirement of delay senstive flow	[0.25,100]ms
PLR requirement of loss sensitive flow	$[10^{-9}, 10^{-8}]$
Throughput requirement of high bitrate flow	[2,10]Mbps
Delay requirement of ordinary flow	[1,10]s
PLR requirement of ordinary flow	$[10^{-5}, 10^{-3}]$
Throughput requirement of ordinary flow	[0.5,3.5]Mbps
Active time	20s

selected path of whole flows.

V. SIMULATION RESULT

In this section, we evaluate the proposed EBQR method by comparing the performance of several existing routing algorithms. To that end, we adopt existing network topology from the Internet Topology Zoo [21] to perform our experiment. Located in New York, the topology consists of 120 nodes and 150 links, as shown in Fig. 2. Also, we implemented SDNC using RYU, which provides APIs that allow us to create control applications [22]. Then, we conduct the simulation on the Mininet emulator, which enables us to create a network of virtual SDFE and links. Specifically, we adopt iperf instruction for performance measurements and the traffic generation at the hosts on Mininet, where OpenFlow protocol is used to communicate with the SDFEs.

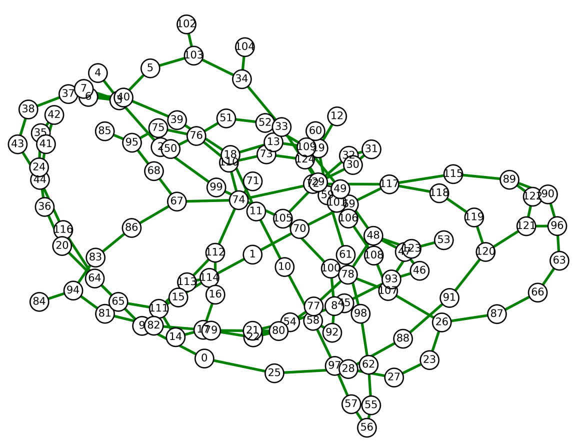

Fig. 2. Network topology of the experimental environment.

In our simulation, we compare the proposed method with *i*) Open Shortest Path First (OSPF), which is a classical routing protocol, *ii*) Lagrangian Relaxation Based Aggregated Cost (LARAC) proposed in [13], which uses the concept of aggregated costs to find the sub-optimal route based on Lagrange relaxation, *iii*) Greedy, which always forward packets to the next hop link with minimal delay. Also, we consider the following performance metrics to evaluate the proposed scheme with the existing algorithms: *i*) Average QoS, which is calculated by the objective of (5), *ii*) Active links, which defined as the number of links used in the network given the set of flows, *iii*) QoS satisfaction ratio, which if the ratio of the number of flows that meet its QoS requirement to the number of whole flow. Furthermore, we consider there are 4 types of flows in the network: delay-sensitive flow, loss-sensitive flow, high bitrate flow and ordinary flow. All of the flows are randomly generated with random source-destination pair. Finally, different simulation parameters are set in Table II as [11], [15].

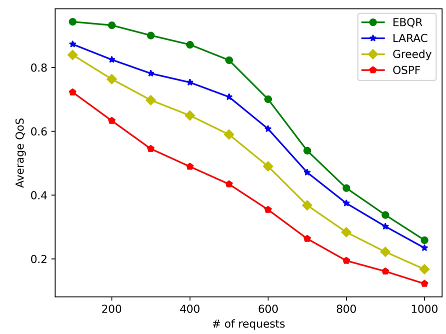

Fig. 3. Average QoS in relation to number of generated request.

To begin with, Fig. 3 illustrates the result of average QoS with respect to the number of generated requests. We can observe that, generally, the average QoS decrease with the number of generated requests. Since the capacity of each link in the edge network is rather limited and is shared by the subset of requests, the more requests exist in the network, the poorer QoS they may receive. Note that all of the algorithms react quite differently before and after # of requests = 600. This is because before requests of 600, the QoS degradation mainly stems from the sharing of resources. However, when # of requests is more than 600, the possibility of flow dropping due to unsatisfiable network conditions is increased dramatically. Though the proposed method outperforms the other existing algorithm under # of requests = 1000, it seems that the QoS performance of all algorithms will decrease to 0.

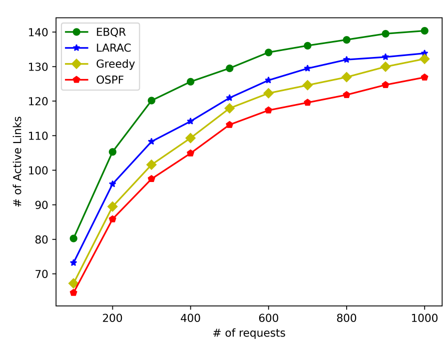

Fig. 4. The number of generated request versus number of active links in the network.

We turn to present the number of active links with respect to the number of generated requests, which can be interpreted as the extent of load balance of the network. We can be seen that in Fig. 4, the number of active links of our method is always more than the other algorithms while maintaining good QoS, as demonstrated in Fig. 3. This can result from the first stage of our algorithm, which is aimed to reach the maximum entropy of the system. Although our main objective is not to achieve load balancing of the network, the entropy-based method indeed can make the routing decision more balanced. On the other hand, since LARAC mainly focuses on satisfying QoS of each flow regardless of load balance, its active links is less than the proposed method. In the end, the number of active links for all of the algorithms will gradually approach 150, which is the number of whole links in our topology.

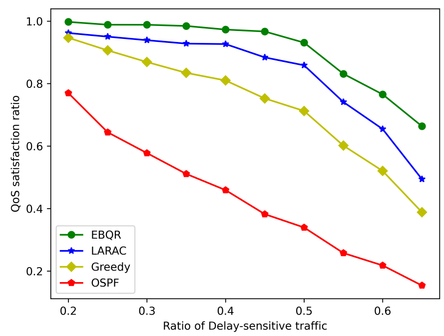

Fig. 5. Qos satisfaction ratio in relation to the ration of dela-sensitive requests in network traffic.

Eventually, Fig. 5 depict the QoS satisfaction ratio versus the ratio of delay-sensitive flows. We can observe that the ratio of QoS satisfaction of all algorithms decreases with the ratio of delay-sensitive flows goes high. This is because the overall network resource can only satisfy QoS requirement for a range of delay-sensitive flows. We can see the proposed method OSPF has the highest and lowest QoS satisfaction ratio as the number of delay-sensitive flows increases. It can result from that OSPF only searches the shortest path for flow without regard to Qos requirement while the proposed method will consider both throughput, delay and PLR. This can show that the SDN-based routing methods are indeed more efficient in serving requests with different QoS requirements than traditional network methods.

VI. Conclusions

In this paper, we investigate the routing problem for QoS guarantee in software-defined edge network with heterogeneous QoS requirement. By considering the effect of link utilization toward packet loss rate and dely, we can determine the more precise routes decision. In this way, we can further enhance the network performance by optimize the QoS of a set of incoming flow while achieving load balance. Also, we proposed a low-complexity heuristic routing algorithm based on the concept of entropy to efficiently optimize the formulated routing problem, which is NP-hard. Simulation results, based on real world network topology, show that the proposed scheme outperforms the OSPF, LARAC and greedy algorithm in terms of QoS, active links and satisfaction ratio.

References

[1] F. Y. Okay and S. Ozdemir, "Routing in Fog-Enabled IoT Platforms: A Survey and an SDN-Based Solution," in IEEE Internet of Things Journal, vol. 5, no. 6, pp. 4871-4889, Dec. 2018.

[2] A. C. Baktir, A. Ozgovde and C. Ersoy, "How Can Edge Computing Benefit From Software-Defined Networking: A Survey, Use Cases, and Future Directions," in IEEE Communications Surveys & Tutorials, vol. 19, no. 4, pp. 2359-2391, Fourthquarter 2017.

[3] S. Agarwal, M. Kodialam and T. V. Lakshman, "Traffic engineering in software defined networks," 2013 Proceedings IEEE INFOCOM, Turin, pp. 2211-2219, 2013.

[4] Y. Guo, Z. Wang, X. Yin, X. Shi and J. Wu, "Traffic Engineering in SDN/OSPF Hybrid Network," 2014 IEEE 22nd International Conference on Network Protocols, Raleigh, NC, pp. 563-568, 2014.

[5] G. Wang, G. Feng, S. Qin and R. Wen, "Efficient traffic engineering for 5g core and backhaul networks," in Journal of Communications and Networks, vol. 19, no. 1, pp. 80-92, February 2017.

[6] C. Ren, S. Bai, Y. Wang and Y. Li, "Achieving Near-Optimal Traffic Engineering Using a Distributed Algorithm in Hybrid SDN," in IEEE Access, vol. 8, pp. 29111-29124, 2020.

[7] S. Xu, X. Wang, G. Yang, J. Ren and S. Wang, "Routing optimization for cloud services in SDN-based Internet of Things with TCAM capacity constraint," in Journal of Communications and Networks, vol. 22, no. 2, pp. 145-158, April 2020.

[8] J. Zhang, M. Ye, Z. Guo, C. Yen and H. J. Chao, "CFR-RL: Traffic Engineering with Reinforcement Learning in SDN," in IEEE Journal on Selected Areas in Communications, to appear.

[9] X. Wang et al., "The Joint Optimization of Online Traffic Matrix Measurement and Traffic Engineering For Software-Defined Networks," in IEEE/ACM Transactions on Networking, vol. 28, no. 1, pp. 234-247, Feb. 2020.

[10] M. Amiri, A. Sobhani, H. Al Osman and S. Shirmohammadi, "SDN-Enabled Game-Aware Routing for Cloud Gaming Datacenter Network," in IEEE Access, vol. 5, pp. 18633-18645, 2017.

[11] N. Saha, S. Bera and S. Misra, "Sway: Traffic-Aware QoS Routing in Software-Defined IoT," in IEEE Transactions on Emerging Topics in Computing.

[12] Q. Fu, E. Sun, K. Meng, M. Li and Y. Zhang, "Deep Q-Learning for Routing Schemes in SDN-Based Data Center Networks," in IEEE Access, vol. 8, pp. 103491-103499, 2020.

[13] H. E. Egilmez, S. Civanlar and A. M. Tekalp, "An Optimization Framework for QoS-Enabled Adaptive Video Streaming Over OpenFlow Networks," in IEEE Transactions on Multimedia, vol. 15, no. 3, pp. 710-715, April 2013.

[14] E. Akin and T. Korkmaz, "Rate-Based Dynamic Shortest Path Algorithm for Efficiently Routing Multiple Flows in SDN," ICC 2019 - 2019 IEEE International Conference on Communications (ICC), Shanghai, 2019.

[15] I. Fajjari, N. Aitsaadi and D. E. Kouicem, "A Novel SDN Scheme for QoS Path Allocation in Wide Area Networks," GLOBECOM 2017 - 2017 IEEE Global Communications Conference, pp. 1-7, 2017.

[16] K. M. Liu, Y. Cao, Y. Liu, G. X. and C. Wu, "A novel min-cost Qos routing algorithm for SDN-based wireless mesh network," 2016 2nd IEEE International Conference on Computer and Communications (ICCC), Chengdu, pp. 1998-2003, 2016.

[17] C. Y. Li, P. k. A. Wai and V. O. K. Li, "Parallel Transfer Optical Packet Switches," in Journal of Lightwave Technology, vol. 27, no. 12, pp. 2159-2168, June15, 2009.

[18] H. Zang, J. P. Jue and B. Mukherjee, "Capacity allocation and contention resolution in a photonic slot routing all-optical WDM mesh network," in Journal of Lightwave Technology, vol. 18, no. 12, pp. 1728-1741, Dec 2000.

[19] M. Shen, H. Chen and N. Xiao, "Entropy-Directed Scheduling for FPGA High-Level Synthesis," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, to appear.

[20] J. Y. Yen, "Finding the K Shortest Loopless Paths in a Network," Management Science, vol. 17, no. 11, pp. 712–716, 1971.

[21] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The internet topology zoo," in IEEE Journal on Selected Areas in Communications,, vol. 29, no. 9, pp. 1765–1775, 2011.

[22] E. Akin and T. Korkmaz, "Comparison of Routing Algorithms With Static and Dynamic Link Cost in Software Defined Networking (SDN)," in IEEE Access,, vol. 7, pp. 148629-148644, 2019.