IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 12, 2018, accepted December 3, 2018, date of publication December 11, 2018,
date of current version January 11, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2886240

Joint Optimization on Bandwidth Allocation and Route Selection in QoE-Aware Traffic Engineering

YI WANG[1], JIAQI ZHENG[2], LIJUAN TAN[2], AND CHEN TIAN[2]
[1]Department of Electronics and Information Engineering, Huazhong University of Science and Technology, Wuhan 430074, China
[2]State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China

Corresponding author: Yi Wang (ywang@hust.edu.cn)

This work was supported in part by the National Science and Technology Major Project of China under Grant 2017ZX03001013-003, in part by the Fundamental Research Funds for the Central Universities under Grant 0202-14380037, in part by the National Natural Science Foundation of China under Grant 61772265, Grant 61602194, Grant 61502229, Grant 61672276, and Grant 61321491, in part by the Collaborative Innovation Center of Novel Software Technology and Industrialization, and in part by the Jiangsu Innovation and Entrepreneurship (Shuangchuang) Program.

ABSTRACT Wide area networks (WANs) are increasingly relying on software-defined networking to orchestrate data transmission. Google and Microsoft built B4 and SWAN, respectively, to interconnect their data centers in WANs, achieving higher network utilization and lower delay, where different priorities can be applied for bandwidth allocation and route selection in their centralized traffic engineering. However, simply relying on the priorities cannot accurately capture the complex properties of the quality of experience (QoE) function corresponding to the end users and always result in a suboptimal solution. We propose QTE, a novel traffic engineering in software-defined WANs, which proactively enforces forwarding policies by coordinating the traffic demand of each data center. The goal of QTE aims to maximize end-user QoE for client-trigger traffic and maximize network throughput for background traffic. We make several technical contributions in designing QTE. First, we formulate maximizing end-user QoE problem as two optimization programs and propose a heuristic algorithm to solve the problem. We further present a concrete design and implementation of the system based on the Floodlight controller. Extensive experiments use Openflow switches on Mininet and numerical simulations, which shows that QTE increases end-user experiences by 45.3% compared with prior work and can be readily implemented on the Floodlight controller.

INDEX TERMS Computer networks, software defined networking.

I. INTRODUCTION

Centralized traffic engineering (TE) is widely used in practice to improve performance of wide area networks (WANs). Increasingly TE is implemented using software defined networking (SDN), where a logically centralized controller maintains a global view of the network state and dispatches TE plans as forwarding rules to the data plane. SDN presents tremendous advantages for data center WANs. Google [1] and Microsoft [2] for example rely on SDN to interconnect their data centers and achieve higher network utilization, lower delay, and less packet drops.

Fig. 1 illustrates the architecture of centralized TE proposed in [2]. Now we briefly review the workflow. The broker first reports the bandwidth demand and priority to the centralized controller, and then the controller determines the allocated bandwidth for this application and sends it to the broker. At the same time, the controller enforces routing policies to the switches in WANs. Finally, the broker limits the sending rate of service hosts according to allocated bandwidth from the controller. Here the priorities are specified based on different network traffic requirements. Basically, the network

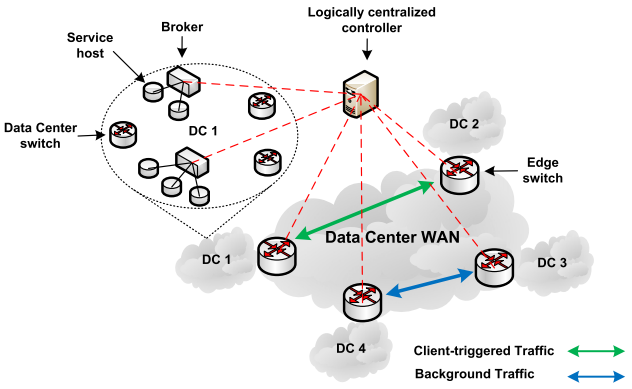


FIGURE 1. Centralized TE in software defined WANs.

TABLE 1. Average RTT between geo-distributed EC2 data centers.

	US West (Oregon)	US East (N. Virginia)	South American (Sao Paulo)	EU (Frankfurt)	Asia Pacific (Singapore)	Asia Pacific (Sydney)
US West (Oregon)	—	95 ms	181 ms	157 ms	162 ms	177 ms
US East (N. Virginia)	94 ms	—	124 ms	109 ms	230 ms	240 ms
South American (Sao Paulo)	181 ms	123 ms	—	216 ms	344 ms	315 ms
EU (Frankfurt)	157 ms	99 ms	218 ms	—	297 ms	324 ms
Asia Pacific (Singapore)	161 ms	231 ms	343 ms	300 ms	—	179 ms
Asia Pacific (Sydney)	178 ms	240 ms	318 ms	321 ms	180 ms	—

We create 6 Amazon EC2 `c4.2xlarge` instances in 6 global data centers in Table 1, each with 8 CPUs and 15 GB memory. We perform over 50 experiments to measure average RTT in EC2 data center WANs.

traffic is divided into two categories [3]: client-triggered traffic and background traffic. Client-triggered traffic is triggered by the customer-facing services such as web search, online chatting and video, etc. Delaying them or bandwidth reduction will directly incur significant performance degradation. Background traffic is largely generated by the applications that perform periodic data back up among geographically distributed data centers, which are elastic to bandwidth allocation and can tolerate the network delay [4]. However, only relying on priority to allocate bandwidth cannot well capture the complexity relationship between end user QoE and the allocated bandwidth and delay, and thus usually incur application performance degradation [5].

In this paper, we propose QTE, a novel traffic engineering that integrates end user QoE properties into centralized routing. Our proposed QTE will significantly benefit the client-trigger traffic that is customer-facing since the QoE metric focuses on *user centric* perspectives. Furthermore, we perform extensive experiments in EC2 data center WANs and the experiment results also demonstrate that the allocated bandwidth and delay in data center WANs play a key role to improve end user QoE.

The contributions of this work are as follows. First, we propose a general optimization framework for QoE-awared traffic engineering in software defined WANs. We formulate Maximizing end user QoE Problem (MQP) as two optimization programs, i.e., how to allocate network bandwidth and delay for client-triggered flows so as to maximize end user QoE. For background flows, our goal is to maximize network utilization. Second, we develop an efficient heuristic algorithm to solve MQP. Finally, we conduct a comprehensive performance evaluation of our algorithm in large-scale network topologies. Simulation results show that our algorithm can improve normalized QoE by 45.3%.

The remainder of the paper is organized as follows. We give a formal definition of MQP in Section II. Section III presents a heuristic algorithm to solve our problem. Experimental evaluation and implementation are presented in Section IV and Section V. Section VI introduces related work and finally Section VII concludes the paper.

II. AN OPTIMIZATION FRAMEWORK

A. A MOTIVATING EXAMPLE

Consider a real application scenario, a user from Oregon attends a meeting in Singapore. When he searched something

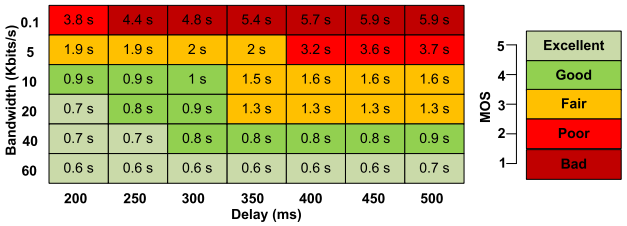


FIGURE 2. MOS with different WAN delay and bandwidth.

from the internet in Singapore, due to TCP performance consideration, the user request is routed to the closed data center located in Singapore. When the search results are finished, the search engine wants to rank the pages based on user's preference (e.g. from user's past social behaviors). Unluckily, the user's preference is located in Oregon data center. At this point, the data center which is processing the user request needs to communicate with the data center which stores the user preference data. With the globally deployment of geo-distributed data center, such type of traffic are increasing significantly.

We perform an experiment at EC2 data center in order to capture the relation between end user QoE and the allocated bandwidth and delay. The average RTT between geo-distributed EC2 data centers is shown in Table 1. We take web browsing as an example, which can be quantified by *page loading time* (PLT) as shown in [6]. To measure the PLT, we consider a single web page generated by the server located in Singapore data center. The user preference is located in Oregon data center. A client sends http request and we use wget tool to measure PLT in client side. The maximum bandwidth (due to the limitation of application server) and minimum delay between these two EC2 data centers are 86Mbits/s and 159ms respectively. We use Linux `TC` command to adjust the value of delay and bandwidth between Oregon and Singapore data center. When we get PLT, we map PLT to a user score by using the ITU Recommendation G.1030 [7] specified for web information retrieval task. We use 5-point MOS (i.e., 5:excellent, 4:good, 3:fair, 2:poor, 1:bad.) as the metric to quantify service performance. The experiment results are illustrated in Fig. 2. We can observe that the allocated bandwidth and delay is a key factor to end user QoE.

Definition 1: Let $U_f(x, y)$ be the QoE function for flow $f \in F_c^{s,t}$, where x and y represent the allocated bandwidth and delay respectively.

From our experiments, we have three observations.

Observation 1: QoE function $U_f(x, y)$ is not continuous.

Observation 2: For client-triggered flow $f \in F_c^{s,t}$, if $x_1 > x_2$, then $U_f(x_1, y) \geq U_f(x_2, y)$.

Observation 3: For client-triggered flow $f \in F_c^{s,t}$, if $y_1 > y_2$, then $U_f(x, y_1) \leq U_f(x, y_2)$.

B. NETWORK MODEL

Before presenting the problem definitions, we first discuss our network model. A network is a directed graph $G = (V, E)$, where V is the set of switches and E the set of links with capacities C_e for each link $e \in E$. From user's perspective [3], the flows can be divided into *client-triggered* flows and *background* flows. $F_c^{s,t}$ represents the set of *client-trigger* flows from ingress switch s to egress switch t. Client-triggered flows such as video do not work well when their flows are split and thus they must only use one tunnel from tunnel set $P^{s,t}$. Each flow $f \in F_c^{s,t}$ is associated with a QoE function $U_f(\cdot)$. $F_b^{s,t}$ represents the set of *background* flows that can be split and use multiple tunnels. The tunnel set $P^{s,t}$ from ingress switch s to egress switch t is pre-computed such that all tunnels are loop-free. The tunnel delay τ_p is a constant in our model. For convenience, we summarize important notations in Table 2.

TABLE 2. Key notations in this paper.

V	The set of switches v
E	The set of links e
G	The directed network graph $G = (V, E)$
C_e	The whole capacity of link e
$C_e^{'}$	The available capacity on link e.
$F_c^{s,t}$	The set of client-triggered flows from ingress switch s to egress switch t
m	The number of client-triggered flows. $m = \|F_c^{s,t}\|$
$F_b^{s,t}$	The set of background flows from ingress switch s to egress switch t
n	The number of background flows. $n = \|F_b^{s,t}\|$
$F^{s,t}$	The set of flows from ingress switch s to egress switch t, $F^{s,t} = F_c^{s,t} \cup F_b^{s,t}$
$P^{s,t}$	The tunnels available from ingress switch s to egress switch t
k	The number of tunnels available from ingress switch s to egress switch t. $k = \|P^{s,t}\|$
τ_p	The delay of path $p \in P^{s,t}$
d_f	The traffic demand for flow $f \in F_b^{s,t}$
U_f	The QoE function for flow $f \in F_c^{s,t}$
$y_{f,p}$	The proportion of flow through tunnel p to the total flow f
x_f	The bandwidth allocated for flow f
w_f	The weight of flow f in the overall flows

C. PROBLEM FORMULATION

Based on the above network model, we formulate Maximizing end user QoE Problem (MQP) as optimization program (1). Given the number of intermediate stages, we wish to find the optimal routing for all intermediate stages that minimizes the transient congestion from the initial stage to the final stage.

Finding the set of tunnels with a given size that carries the most traffic is NP-hard [8]. The set cover is NP-hard [8],and the classic set cover problem can reduced to the special case of MQP, the general MQP is NP-hard. We use the following heuristic: For each ingress egress switch pair, we compute a tunnel set P of k different tunnels (k shortest or almost disjoint) over which the traffic can be routed. For certain switch, if its rule space can't accommodate so many tunnels, we delete some tunnels. We randomly select a tunnel for deletion at regular intervals, similar to the approach taken in SWAN [2], until all the switches in the network can work for all tunnels. When the tunnels are allocated, we need to split each aggregate flow across possible tunnels.

The goal of client-triggered traffic is to maximize end user experience and the formulation is defined as follows:

$$\text{maximize} \sum_{f \in F_c^{s,t}} \sum_{p \in P^{s,t}} y_{f,p} \cdot U_f(x_f, \tau_p) \tag{1}$$

$$\text{subject to} \sum_{f \in F_c^{s,t}} \sum_{p \in P^{s,t}: e \in p} x_f \cdot y_{f,p} \leq C_e, \quad \forall e \in E, \tag{1a}$$

$$\sum_{p \in P^{s,t}} y_{f,p} = 1, \quad \forall f \in F_c^{s,t}, \tag{1b}$$

$$y_{f,p} \in \{0, 1\}, \quad \forall f \in F_c^{s,t}, \forall p \in P^{s,t}, \tag{1c}$$

$$x_f \geq 0, \quad \forall f \in F_c^{s,t}. \tag{1d}$$

The objective of (1) is to maximize QoE function, which reflects end user QoE. The optimization variables $y_{f,p}$ indicate whether flow f is routed through tunnel p. The delay of tunnel p is τ_p. Constraint (1a) is the link capacity constraint, where the left side represents the load in link e. Constraint (1b) is the flow demand conservation constraint. Constraint (1c) is the integer constraint representing one flow can only be routed to one tunnel.

The goal of QTE for background flows is to maximize weighted network throughput. We formulate maximizing weighted network throughput problem (MWP) as linear program (2).

$$\text{maximize} \sum_{f \in F_b^{s,t}} \sum_{p \in P^{s,t}} w_f \cdot x_f \cdot y_{f,p} \tag{2}$$

$$\text{subject to} \sum_{f \in F_b^{s,t}} \sum_{p \in P^{s,t}: e \in p} x_f \cdot y_{f,p} \leq C_e', \quad \forall e \in E \tag{2a}$$

$$\sum_{p \in P^{s,t}} y_{f,p} = 1, \quad \forall f \in F_b^{s,t}, \tag{2b}$$

$$y_{f,p} \geq 0, \quad \forall f \in F_b^{s,t}, \forall p \in P^{s,t}, \tag{2c}$$

$$x_f \geq 0, \quad \forall f \in F_b^{s,t}. \tag{2d}$$

III. AN EFFECTIVE HEURISTIC ALGORITHM

The main intuition is that, we firstly make all flows achieve the required rate d_f, in such case, the total utility in the

network is maximum and the utility loss is zero. If the rate assignment of all the flows satisfy the link capacity constraint, this would be undoubtedly the optimal solution. If there exist some links whose flow rate is beyond its capacity, we decrease the allocated rate for each flow according to the curve of utility function until the rate in all links satisfy the capacity constraint. It's obvious that we should firstly decrease the flow rate whose derivative of utility function is small. Our detailed heuristic algorithm is described in Algorithm 1.

Algorithm 1 QTE Algorithm

Input: Network topology $G = (V, E)$; link capacity C_e; tunnel set $P^{s,t}$ from ingress switch s to egress switch t; the QoE function $U_f(\cdot)$ for $f \in F_c^{s,t}$; step Δ.

Output: The solution $\{y_{f,p}\}$ and $\{x_f\}$ for $f \in F_c^{s,t}$ and $f \in F_b^{s,t}$

1: Calculate the stable point (d_f, τ_p) of QoE function $U_f(\cdot)$ for $f \in F_c^{s,t}$
2: Apply Algorithm 2 to obtain the solution $\{y_{f,p}\}$ and $\{x_f\}$ for $f \in F_c^{s,t}$
3: Update available bandwidth C'_e for $e \in E$
4: The optimal solutions $\{y_{f,p}\}$ and $\{x_f\}$ for $f \in F_b^{s,t}$ of problem (1) can be obtained in polynomial time using standard solvers.

Algorithm 2 Iterative Algorithm

Input: Network topology $G = (V, E)$; tunnel set $P^{s,t}$ for application f; utility function $U_f(\cdot)$ for $f \in F$; step Δ.

Output: The solution $\{y_{f,p}\}$ and $\{x_f\}$ to (1)

1: **for** each source destination switch pairs (s, t) **do**
2: **for** each path p of destination switch pairs (s, t) **do**
3: find max $\frac{e \in p C_e}{p \in P^{s,t} \tau_p}$
4: **end for**
5: $\sigma = \sum_{p \in P^{s,t}} \phi(p)$, where $\phi(p) = arg\,min_{\,e \in p} C_e$.
6: $d = \sum_{f \in F_c^{s,t}} d_f$
7: **while** d σ **do**
8: $\hat{f} = arg\,min_{f \in \hat{F}} \frac{U_f(x_f, \tau) - U_f(x_f - \Delta, \tau)}{\Delta}$
9: $d_{\hat{f}} = d_{\hat{f}} - \Delta$
10: Update d
11: **end while**
12: **end for**

We run Algorithm 1 and calculate a point (d_f, τ_p) to make function $U_f(\cdot)$ maximum (line 1). If that point cannot do it, then we will apply Algorithm 2 to adjust the point, until it meets the requirements in the link. Then we get the $\{y_{f,p}\}$ and $\{x_f\}$ for $f \in F_c^{s,t}$ (line 2). Because the bandwidth in link e is occupied by some flows, then update available bandwidth for $e \in E$(line 3). For $f \in F_b^{s,t}$ the solutions can be obtained in polynomial time using standard solvers. So don't need the heuristic algorithm which we proposed to get solutions.

We first run Algorithm 2 and get a path p to maximize the ratio of C_e to τ_p (line 2,3). And then we calculate C_e for each link. Based on it, we pick the link e with minimal link capacity (line 5). For flows routed through link e, we construct flow set $\hat{F}$ (line 6). For each flow, we decrease each flow's rate until the total rate is less than or equal to link capacity (lines 7-11). Specifically, in each iteration, we calculate the variation of utility function relative to Δ (line 8). Note that this formula can handle the case even though the utility function is not continuous such as hard-deadline applications in WAN. If the total rate in each link is not beyond its capacity, the algorithm stops. Note that if the utility function is $U_f(x) = x$ for each $f \in F$, the objective of problem (1) is actually to maximize network throughput.

IV. EXPERIMENTAL EVALUATION

We conduct extensive experiments to evaluate our algorithms in this section.

A. SETUP

We consider synthetic topology and realistic topology respectively in our evaluation.

- A synthetic scale-free topology randomly produced by the `scale_free_graph` function in [9], which is referred to as ScaleFree topology. The capacity of all links is set to be 100 Gbps.
- A realistic topology for interconnecting Microsoft's data center WANs [10], which is illustrated in Fig. 6(a). The capacity of all links is set to be 100 Gbps.

We consider tunnel based routing [2] in our scenarios. The flows in the network are generated randomly, and we change the flow demand to simulate traffic variations.

B. BENCHMARK SCHEMES

We evaluate the following schemes.

QTE: Our proposed algorithm as shown in Algorithm 1.

SWAN: SWAN [2] takes advantage of different priorities to address bandwidth allocation and routing problem. The results can be obtained by solving a sequence of LPs. This method does not take end user QoE into consideration. Thus it cannot be used to solve our program (1).

MMF: The bandwidth allocation method using max-min fairness. If and only if the allocation is feasible, if the increase of one party's allocation will inevitably lead to the reduction of the other party's allocation. That is max-min fairness.

C. BASIC PERFORMANCE

QoE has a wide range of values, and normalized QoE maps values to between 0 and 1.We first investigate the normalized QoE for different schemes. Fig. 3 shows that the normalized QoE for QTE, SWAN and MMF in both Microsoft topology and scale-free topology. As the number of flows become large, the normalized QoE decreases in all cases. The reason is that the limited network capacity cannot meet the bandwidth requirements of all flows. On average, we can see that the normalized QoE for QTE can be increased by 20% and 40% compared with SWAN and MMF. Specifically, when

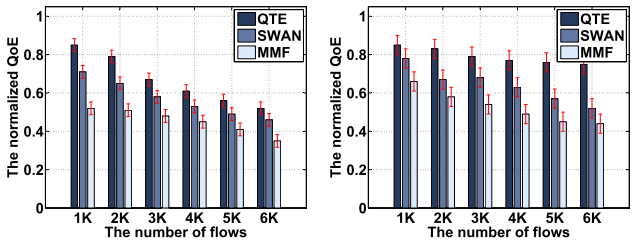


FIGURE 3. The normalized QoE for different schemes. (a) Microsoft topology. (b) ScaleFree topology.

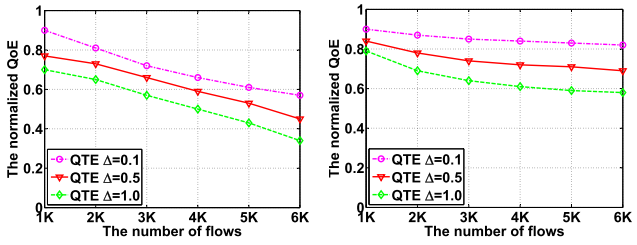


FIGURE 4. The normalized QoE for different parameter Δ. (a) Microsoft topology. (b) ScaleFree topology.

the number of flows is 1K as shown in Fig. 3(a), the normalized QoE for QTE, SWAN and MMF is 0.83, 0.7 and 0.5, respectively.

Fig. 4(b) shows that the normalized QoE for different parameter settings for Δ in Microsoft topology and scale-free topology. Both figures show that the normalized QoE reduces as the parameter Δ increases, which indicate a large value for Δ can lead to more QoE reduction. The QoE in Fig. 4(a) drops linearly and has a larger reduction compared with the results in Fig. 4(b).

D. IMPLEMENTATION IN FLOODLIGHT

We develop a prototype of our algorithms using Floodlight 1.0 controller [11] and evaluate the performance in Mininet 2.0 [12]. The forwarding rules are installed and updated via Floodlight's REST API. Our hardware configuration is PC: Intel i5-2400 Quad-core processor.

Mininet Setup: We consider two realistic topologies in Mininet shown in Fig. 6. Specifically,

- A realistic WAN topology for interconnecting Microsoft's data centers [10], which is illustrated in Fig. 6(a). There are 8 switches and 14 100Mbps links.
- NetRail [13] is publicly service provider network topology, as illustrated in Fig. 6(b). There are 7 switches and 10 100Mbps links.

Table 3 shows how background flows use multiple tunnels in our implementation in the ingress switch S_2 and egress switch S_7 for the WAN shown in Fig. 6(b) as an example. The action of S_2 points to the group table Gr 1.1 of type `select`. Gr 1.1 performs multipath routing over four tunnels, and stamps packets with four VLAN IDs. Egress switch

TABLE 3. Flow table and group table at ingress switch S_2 and egress switch S_7 for splittable flows in Fig. 6(b).

Flow table at S_2

Matching Field				Action
InPort	SrcPfx	DstPfx	Tag	
host 1	—	—	—	Gr 1.1

Group table at S_2

Identifier	Type	Action Buckets
Gr 1.1	Select	Weight: 3; Push vlan(0x200); Output: link $\langle S_2, S_1\rangle$ Weight: 1; Push vlan(0x201); Output: link $\langle S_2, S_5\rangle$ Weight: 1; Push vlan(0x202); Output: link $\langle S_2, S_4\rangle$ Weight: 1; Push vlan(0x203); Output: link $\langle S_2, S_6\rangle$

Flow table at S_7

Matching Field				Action
InPort	SrcPfx	DstPfx	Tag	
—	—	10.0.0.2	0x200	Pop vlan; Output: host 2
—	—	10.0.0.3	0x200	Pop vlan; Output: host 3
—	—	10.0.0.4	0x201	Pop vlan; Output: host 4
—	—	10.0.0.5	0x201	Pop vlan; Output: host 5
—	—	10.0.0.6	0x202	Pop vlan; Output: host 6
—	—	10.0.0.7	0x202	Pop vlan; Output: host 7
—	—	10.0.0.8	0x203	Pop vlan; Output: host 8
—	—	10.0.0.9	0x203	Pop vlan; Output: host 9

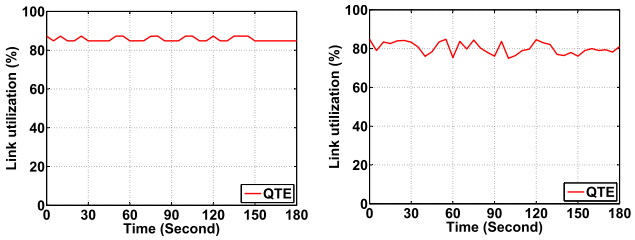


FIGURE 5. The link utilization in Mininet. (a) Microsoft topology. (b) NetRail topology.

S_7 first pops the VLAN header and then forwards the packet according to its destination IP.

Fig. 5 shows the changes in time utilization of the link. We use Floodlight statistic module to measure the maximum link utilization $\mu_{\Delta t}$ in the network every five seconds, i.e., $\Delta t = 5$s.

$$\mu_{\Delta t} = \max\left\{\frac{L_{e,\Delta t}}{C_e}\middle| e \in E\right\}$$

where the parameters $L_{e,\Delta t}$ and C_e represent the link load and link capacity at link e. In Fig. 5(a), we can observe that the link utilization fluctuates slightly and its value can beyond 80% in the Microsoft topology. The same results can be also shown in Fig. 5(b). The QTE can in general lead to high link utilization.

V. RELATED WORK

There is a rich literature on traffic engineering for IP networks [1], [14], [15] and data center WAN [1], [2], [16]. While several of these prior approaches maximize network throughput or minimize of maximum link utilization. B4 [1] and SWAN [2] support prioritization of different traffic classes, but priority alone cannot capture the non-convexity

FIGURE 6. Two realistic network topologies used in our evaluation. (a) Microsoft's inter-data center WAN topology. (b) NetRail topology.

of delay in applications' utility functions. FUBAR [17] takes bandwidth utility and delay utility as optimization objective, but it doesn't consider SP's revenues and the failure case.

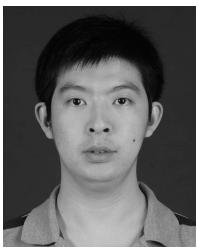
VI. CONCLUSION

In this paper, we studied the problem of maximizing end user QoE. We formulated it as two optimization programs and proposed a heuristic algorithm to solve the problem. Experimental and simulation results show that our algorithms can improve the QoE and be compatible with existing Floodlight controller.

REFERENCES

[1] S. Jain *et al.*, "B4: Experience with a globally-deployed software defined WAN," in *Proc. SIGCOMM*, 2013, pp. 3–14.
[2] C.-Y. Hong *et al.*, "Achieving high utilization with software-driven WAN," in *Proc. SIGCOMM*, 2013, pp. 15–26.
[3] Y. Chen, S. Jain, V. K. Adhikari, Z. L. Zhang, and K. Xu, "A first look at inter-data center traffic characteristics via Yahoo! datasets," in *Proc. INFOCOM*, 2011, pp. 1620–1628.
[4] N. Laoutaris, M. Sirivianos, X. Yang, and P. Rodriguez, "Inter-datacenter bulk transfers with netstitcher," in *Proc. SIGCOMM*, 2011, pp. 74–85.
[5] O. Hohlfeld, E. Pujol, F. Ciucu, A. Feldmann, and P. Barford, "A QoE perspective on sizing network buffers," in *Proc. IMC*, 2014, pp. 333–346.
[6] S. Egger, T. Hoßfeld, R. Schatz, and M. Fiedler, "Waiting times in quality of experience for Web based services," in *Proc. QoMEX*, 2012, pp. 86–96.
[7] *Estimating End-to-End Performance in IP Networks for Data Applications*, Standard ITU-T G.1030, 2005.
[8] T. Hartman, A. Hassidim, H. Kaplan, D. Raz, and M. Segalov, "How to split a flow?" in *Proc. INFOCOM*, 2012, pp. 828–836.
[9] *Networkx*. Accessed: Dec. 2018. [Online]. Available: https://networkx.github.io/
[10] X. Jin *et al.*, "Dynamic scheduling of network updates," in *Proc. SIGCOMM*, 2014, pp. 539–550.
[11] *Floodlight*. Accessed: Dec. 2018. [Online]. Available: http://floodlight.openflowhub.org/
[12] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: Rapid prototyping for software-defined networks," in *Proc. HotNets*, 2010, p. 19.
[13] *Topology Zoo*. Accessed: Dec. 2018. [Online]. Available: http://topology-zoo.org/
[14] S. Kandula, D. Katabi, B. S. Davie, and A. Charny, "Walking the tightrope: Responsive yet stable traffic engineering," in *Proc. SIGCOMM*, 2005, pp. 253–264.
[15] H. Wang, H. Xie, L. Qiu, Y. R. Yang, Y. Zhang, and A. Greenberg, "COPE: Traffic engineering in dynamic networks," in *Proc. SIGCOMM*, 2006, pp. 99–110.
[16] S. Kandula, I. Menache, R. Schwartz, and S. R. Babbula, "Calendaring for wide area networks," in *Proc. SIGCOMM*, 2014, pp. 515–526.
[17] N. Gvozdiev, B. Karp, and M. Handley, "FUBAR: Flow utility based routing," in *Proc. HotNets*, 2014, pp. 1–12.

YI WANG received the B.S., M.S., and Ph.D. degrees from the Department of Electronics and Information Engineering, Huazhong University of Science and Technology, China, in 2000, 2003, and 2009, respectively. She is currently a Lecturer with the School of Electronics Information and Communications, Huazhong University of Science and Technology. Her research interests include cloud computing.



JIAQI ZHENG was a Research Assistant with the City University of Hong Kong, in 2015, and was a Visiting Scholar with Temple University, in 2016. He is currently an Assistant Researcher with the Department of Computer Science and Technology, Nanjing University, China. His research interests include data center networks, software-defined networks, and cloud computing. He has produced a number of high-quality papers in journals, including the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, the IEEE TRANSACTIONS ON SERVICES COMPUTING, the IEEE ICNP, the IEEE ICDCS, the IEEE ICPP, and the IEEE SECON. He was a recipient of the Best Paper Award from the IEEE ICNP 2015.

LIJUAN TAN received the B.Eng. degree from the School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. She is currently pursuing the master's degree with Nanjing University. Her research interest includes data center networks.

CHEN TIAN received the B.S., M.S., and Ph.D. degrees from the Department of Electronics and Information Engineering, Huazhong University of Science and Technology, China. He was an Associate Professor with the School of Electronics Information and Communications, Huazhong University of Science and Technology. From 2012 to 2013, he was a Postdoctoral Researcher with the Department of Computer Science, Yale University. He is currently an Associate Professor with the State Key Laboratory for Novel Software Technology, Nanjing University, China. His research interests include data center networks, network function virtualization, distributed systems, Internet streaming, and urban computing.