SDN-based Multi-Class QoS-guaranteed Inter-Data Center Traffic Management

Jason Min Wang, Ying Wang, Xiangming Dai, Brahim Bensaou
Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
{jasonwangm, ywangbf, xdai, brahim}@cse.ust.hk

***Abstract*—When allocating network bandwidth to multiple classes of applications in inter-data center communication, coordination always yields a better utilization of the backbone network. Yet, it often comes at a prohibitively heavy computational and communication cost, making it thus far not a practically viable approach. SDN helped in bridging the communication cost gap by enabling centralized control, and SDN has been recently applied in such inter-DC traffic management. However, the computational cost is still an issue as the efficient and fast response of the centralized traffic engineering algorithm has become crucial to the practicality of such SDN-based approach. In this paper, we present** MCTEQ**, a utility-optimization-based joint-bandwidth allocation for inter-DC communication with multiple traffic classes, that handles priorities between traffic classes in a soft manner and explicitly considers the delay requirement of** Interactive **flows.** MCTEQ **being NP-hard, we apply approximation techniques to lean on the mature and efficient LP solver and obtain fast and accurate approximations. We demonstrate via experiments with Google's inter-DC backbone topology that** MCTEQ **achieves about 160 Gbps higher network utilization than the existing** SWAN **solution, yet runs 2.5 times faster. In particular,** MCTEQ **guarantees that the allocated bandwidth for** Interactive **flows strictly meets their end-to-end delay requirements.**

Keywords*—*SDN, quality of service, traffic management

I. Introduction

Recent years have witnessed an unprecedented growth in the number of data centers (DCs) being built by large cloud service providers (CSPs). To provide flexible and reliable services at the global scale, CSPs have deployed multiple DCs at different geographical areas, often spanning continents, and interconnect them via private high-speed backbone networks, offering hundreds of Gbps or tens of Tbps bandwidth. The backbone network is a critical infrastructure for a CSP, since many services rely on low-latency inter-DC communication and packet losses are typically considered unacceptable. For this reason, backbone links are often over-provisioned with 2-3x bandwidth. In spite of the heavy cost, these links often have a poor utilization, 30-50% on average [1]. Therefore, it is desirable to improve the efficiency of inter-DC backbone networks while maintaining the quality of service of the traffic they carry.

The inefficiency of current inter-DC backbone network stems from three aspects. First, the lack of effective control techniques cannot make efficient use of the network resources. With no coordination, each application or service now can send however much traffic whenever it wants, in total oblivion to the current network load. As a result, the bandwidth needs to be over-provisioned for the network to be able to digest the superimposed peak demand. In reality, with a little coordination, the demand for bandwidth could be reduced by postponing the delivery of delay-tolerant services to off-peak periods. Second, the widely varying performance requirements of applications are usually ignored. For example, interactive applications (e.g., web search) are delay-sensitive, while background applications (e.g., data synchronization between data centers) are throughput-sensitive. Third, it is known that traffic engineering with traditional distributed routing protocols (e.g., link state) is suboptimal in most cases [2]. Distributed approaches are often inflexible and hardly lend themselves to the deployment of sophisticated resource sharing principles such as fair bandwidth sharing among services with priorities or multi-path forwarding to balance application traffic in response to link failures and demand changes.

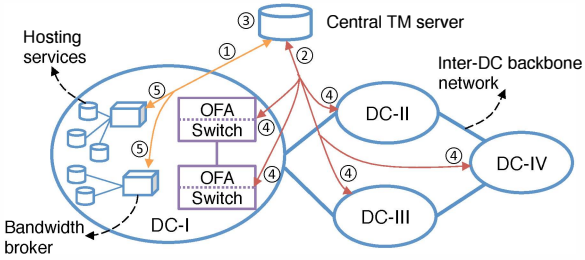


Fig. 1. SDN-based traffic engineering for inter-DC backbone networks.

SDN-based approach. The emerging software-defined networking (SDN) technique has recently been used for inter-DC traffic management [1], [3], to conquer the above mentioned inefficiencies of traditional approaches. Fig. 1 briefly explains how traffic management (TM) for the inter-DC backbone network is done in a centralized way. The *central TM server*, as a logical controller, serves to coordinate the network activities of applications. Traffic management is done periodically, e.g., every 5 minutes. At the beginning of each period, the *bandwidth broker* estimates the bandwidth demands of applications for the current period and reports the aggregate information to the TM server (in step ①). The Open-flow enabled *OFA Switch* reports the network events and traffic statistics to the central TM server (in step ②). This enables the TM server to maintain the global network state information. Based on the collected bandwidth demands and the network state, the TM server computes the bandwidth allocation for the competing applications, possibly using multiple paths (in step ③). This centralized approach is flexible to enable various traffic engineering goals, which is the focus of this paper. The results of traffic engineering consist of two parts: the tunnel settings, and the rate limit enforcement

978-1-4799-2730-2/14/$31.00 ©2014 IEEE

on each tunnel. Tunnels represent routing paths and can be implemented using either IP in IP encapsulation or MPLS. To achieve high network utilization, new tunnels must be set up and some existing tunnels must be torn down periodically (in step ④). This involves changing the forwarding states in the OFA switches. The allocated bandwidth for applications will be rate limited by the bandwidth broker (in step ⑤), possibly using token buckets or other means. With the appropriate admission control, quality of service (QoS) requirements (e.g., bandwidth, and end-to-end delay) can be satisfied.

Challenges and motivations. The fundamental goal of inter-DC traffic engineering is to allocate bandwidth for applications over paths, while achieving two goals: maximizing the network utilization, and taking fairness and QoS requirements into consideration. Turning the SDN-based approach into practice entails addressing the following challenges. First, the data center traffic falls into three classes (i.e., background, elastic, and interactive), each having different performance requirements. In particular, interactive flows require end-to-end delay guarantee. Second, existing hardware switches support only a limited number of forwarding rules which are used to set up tunnels; yet fully utilizing the network often demands a large number of candidate routing paths. Thus, we need to strike a balance between the available candidate set of routing paths and the network utilization. Third, we need a scalable and fast algorithm, since the traffic management is carried out periodically with relatively short periods. An algorithm with a long execution time (e.g., >10 seconds), no matter how "good" the results it yields, is just not good enough. Fourth, the forwarding state updates (in step ④ of Fig. 1) involve multiple distributed switches and need to be done in a consistent and congestion-free manner. This challenging problem has been well studied in [3]–[6] and feasible solutions have been proposed therein. Thus, what remains to achieve in this area to bring such SDN based solution to fruition is to design fast and accurate approximations that enable the deployment of periodic traffic engineering in inter-DC networks.

Contributions. This paper makes four main contributions. First, we formulate the MCTEQ problem that adopts and advocates a joint bandwidth allocation to all traffic classes in the network instead of using the strict priority alternative approach (simpler but suboptimal). To this end, we use utility functions with different weights to reflect priorities of traffic classes in allocating bandwidth. This differs from SWAN [3] where bandwidth allocation is done step-by-step in a strict priority order, i.e., allocating bandwidth to traffic class with the highest priority first. Furthermore, to enable intra-class fairness, the concave log utility function in MCTEQ enforces the proportional fairness principle among applications within the same traffic class. Second, we explicitly take into account the end-to-end delay requirement of interactive applications. Third, MCTEQ is a multi-class QoS-guaranteed inter-DC traffic management problem and is NP-hard. We use piece-wise linear functions to approximate the log utility function and transform the bilinear term in the end-to-end delay constrain into linear constraints with binary variables. The resulting transformed problem makes the "branch-and-bound with LP" solution very highly computationally efficient. Fourth, we present an extensive evaluation over a realistic inter-DC backbone network topology and show that MCTEQ outperforms existing solutions.

II. Multi-Class QoS-Guaranteed Traffic Management

We first describe the inter-DC traffic classes; then present the inter-DC traffic management model MCTEQ, followed by the approximating technique that enables us to obtain a fast and efficient solution. Finally, we briefly discuss related work.

A. Traffic Classes

The inter-DC backbone network carries traffic from applications and services hosted in the CSP's cloud. For instance, client requests for front-end services (e.g., web search, photo sharing) rely on a composition of sub-services at two or more data centers. It is not surprising that different applications and services exhibit distinct performance requirements. In general, they are categorized into three broad classes:

- *Interactive traffic.* It includes services that are closely tied with end-user experience, manifested by a real-time requirement. For example, in response to a user request, the associated data needs to be copied from one data center to another. For interactive services, even small increases in the response time (100 ms) could lead to great business loss. Thus, interactive traffic is highly sensitive to delay and packet losses.
- *Elastic traffic.* It includes services that are less critical to end-user experience, but still require timely delivery. For example, some distributed computational jobs, like Map-Reduce jobs, demand remote access to data stored on other data centers. Elastic traffic is less delay sensitive and should be delivered within a few seconds to a few minutes.
- *Background traffic.* It includes various routine maintenance tasks, like the large-scale data synchronization among data centers. Typically, background services demand a large bandwidth, but are insensitive to delay. The traffic delivery is expected to complete within a few minutes to a few hours.

In summary, interactive traffic occupies the smallest volume and is the most sensitive to delay; background traffic occupies the largest volume and is the least sensitive to delay. The above three traffic classes are ordered in decreasing latency sensitivity and decreasing overall priority.

B. Centralized Traffic Management Model – MCTEQ

We model the inter-DC backbone network as a digraph $\mathcal{G} = (\mathcal{I}, \mathcal{L})$, where $\mathcal{I}$ is the set of data centers and $\mathcal{L}$ is the set of unidirectional links. To support large volumes of data transfer between remote data centers, the long-haul links connecting DCs typically have high capacities. Let c_l be the link capacity and p_l be the link propagation delay. We use $\mathcal{K} = \{\text{Int}, \text{Ela}, \text{Bac}\}$ to represent the three traffic classes.

Flow group. Traffic within a given class can belong to several applications, each of which in turn may generate multiple flows. For scalability purpose, traffic management cannot work at the fine granularity level of the individual flow or application. Thus, flows are aggregated into flow groups (FG) defined by the tuple $(\text{source}, \text{destination}, \text{class})$. For example, (i_1, i_2, Int) represents the FG including all interactive flows from DC i_1 to DC i_2. For each traffic class $k \in \mathcal{K}$, $\mathcal{F}_k$ is the set of all corresponding FGs. r_f is the aggregate bandwidth demand of FG f, estimated by the bandwidth broker. For notational convenience, denote $\mathcal{F}$ as the set of all flow groups.

Tunnel and routing matrix. A tunnel represents a site-level path connecting two DCs in the network, e.g., a sequence of DCs (A→B→C). Let $\mathcal{P}_f$ be the set of candidate paths for FG f. For scalability purpose, $\mathcal{P}_f$ does not necessarily contain all possible paths between source site and destination site in the physical topology. Denote by $\mathcal{P}$ the union of all tunnels for all FGs. The routing matrix $\mathbf{R}$, of size $|\mathcal{P}| \times |\mathcal{L}|$, encodes the mapping between tunnels and links: $R_{p,l}$ is 1 if tunnel p uses link l and is 0 otherwise.

Decision variables. Define $\{x_{f,p} | f \in \mathcal{F}, p \in \mathcal{P}_f\}$ as continuous decision variables that represent the portion of FG-f's demand that will be carried over path p. Let $x_f = \sum_{p \in \mathcal{P}_f} x_{f,p}$ be the aggregate portion of the allocated demand, i.e., $x_f r_f$ corresponds to the actual bandwidth allocated to FG f.

Utility-based joint bandwidth allocation. When allocating bandwidth, interactive traffic has the highest priority, while background traffic has the lowest priority, i.e., $\text{Interactive} > \text{Elastic} > \text{Background}$. The priorities serve to coordinate the actions of applications and take their performance requirements into consideration. SWAN [3] handled priorities in a step-by-step manner, i.e., allocating bandwidth for the three classes of traffic separately following the strict priority order. As such, the effect of the higher priority traffic on the lower priority traffic simply amounts to a constant decrease in the network capacity, and albeit simple, this approach may reduce the overall bandwidth utilization, as it ignores the benefits of statistical multiplexing. In contrast, we advocate a unified approach (via a joint optimization) where the allocation for higher-priority traffic on candidate routes should be balanced optimally to allow for the maximum utilization of network resources by the competing lower-priority traffic. We adopt the log utility function to achieve proportional-like fairness among FGs within the same class. Moreover, the priorities are reflected via the weights associated with the FGs utilities. The utility functions of the three traffic classes are defined as $U_k(x) = w_k \log(1+x)$, where x is the allocated bandwidth in units of Mbps, and w_k is the weight of FG k. In this paper, we set $w_{\text{Int}} = 1000$, $w_{\text{Ela}} = 50$, and $w_{\text{Bac}} = 1$.

QoS guarantee for interactive flows. Interactive applications require a bounded delivery delay. The end-to-end delay on a given path is the sum of propagation delays and queueing delays for all the links on the path. While the propagation delay is easy to obtain, the queueing delay experienced by a flow on a given physical link depends on several factors, including the bandwidth allocated on that link, and the link scheduling policy. Therefore, to bound the path queueing delay for interactive flows, we need to choose the service discipline along the path first. Several service disciplines have been proposed in the literature, and we focus on guaranteed rate service disciplines such as WFQ [7] and other similar more efficient approximations as they are known to provide each flow with a minimum guaranteed rate of service independently of the traffic characteristics of other flows on the path. In conjunction with the appropriate admission control policies (e.g., token bucket), such disciplines guarantee in particular a bounded end-to-end delay [7].

Using the notation in Table I, assume that the service discipline on all the hops along path p of a FG f belong to a class of guaranteed rate (GR) schedulers [8], [9]. Then it is sufficient that FG f traffic conforms to a token bucket with parameters $(\sigma_f, x_{f,p} r_f)$ for the end-to-end delay d_f^j of packet p_f^j, to be bounded by:

$$d_{f,p}^j \leq \left[\sigma_f + (n_p - 1) \max_{1 \leq i \leq j} \ell_f^i\right] / (x_{f,p} r_f) + \sum\nolimits_{n \in p} \beta^n + \tau_p, \quad (1)$$

where β^n is the scheduling constant at router n and τ_p is the propagation delay of path p. For illustration, we use WFQ whose scheduling constant β^n is $\frac{L_n^{\max}}{C^n}$, and $L_n^{\max}$ is the maximum length of packet served by router n. For simplicity, we assume that $\ell_f^{\max} = L_n^{\max} = \ell^{\max}$. Then, the end-to-end delay bound for any packet in FG f on path p under WFQ scheduling discipline, denoted by $d_{f,p}$, is given by:

$$d_{f,p} = \left[\sigma_f + (n_p - 1)\ell^{\max}\right] / (x_{f,p} r_f) + \tau_p + \beta_p \quad (2)$$

where $\beta_p = \sum_{l \in p} \frac{\ell^{\max}}{c_l}$. It should be noted that the delay bound for a FG also applies to the individual flows within the FG [9].

TABLE I. SYMBOLS FOR BOUNDING END-TO-END DELAY

Symbol	Definition
β^n	the scheduling constant at router n
β_p	the accumulated scheduling constants of routs in path p
τ_p	the total propagation delay of path p
n_p	number of routers on path p
ℓ_f^i	the length of packet p_f^i, the i-th packet of flow group f
$\ell_f^{\max}$	the maximum packet length in flow group f
$L_n^{\max}$	the maximum packet length at router n
$d_{f,p}$	the end-to-end delay bound for any packet in flow group f on path p
$\bar{d}_f$	the tolerable end-to-end delay for any packet in flow group f

End-to-end delay constraint for interactive FGs. To explicitly enforce the delay requirement for interactive traffic, we introduce a boolean variable $y_{f,p} \in \{0,1\}$, indicating whether FG $f \in \mathcal{F}_{\text{Int}}$ uses the path p. This is necessary because the delay constraint $d_{f,p} \leq \bar{d}_f$ exists only if a positive fraction of traffic of FG f is routed on p. That is, only if $x_{f,p} > 0$, then $d_{f,p} \leq \bar{d}_f$ has to be satisfied. This condition is fully characterized by the following three constraints: (i) $y_{f,p} \geq x_{f,p}$; (ii) $y_{f,p} \in \{0,1\}$; and (iii) $y_{f,p}(d_{f,p} - \bar{d}_f) \leq 0$. For notational convenience, let $u_{f,p} = \sigma_f + (n_p - 1)\ell^{\max}$ and $v_{f,p} = (\bar{d}_f - \tau_p - \beta_p) r_f$. Note that $u_{f,p}$ and $v_{f,p}$ are FG-path dependent constants. Then, the last constraint (iii) becomes $u_{f,p} y_{f,p} - v_{f,p} x_{f,p} y_{f,p} \leq 0$.

MCTE with QoS guarantee (MCTEQ). With end-to-end delay guarantee for interactive flows, the inter-DC traffic management problem is formulated as:

MCTEQ:

$$\max_{\mathbf{x},\mathbf{y}} \ \sum\nolimits_{k \in \mathcal{K}} \sum\nolimits_{f \in \mathcal{F}_k} U_k\Big(r_f \cdot \sum\nolimits_{p \in \mathcal{P}_f} x_{f,p}\Big) \quad (3a)$$

$$\text{s.t.} \ \sum\nolimits_{p \in \mathcal{P}_f} x_{f,p} \leq 1, \ \forall f \in \mathcal{F} \quad (3b)$$

$$\sum\nolimits_{f \in \mathcal{F}} \sum\nolimits_{p \in \mathcal{P}_f} R_{p,l} x_{f,p} r_f \leq c_l, \ \forall l \in \mathcal{L} \quad (3c)$$

$$x_{f,p} \geq 0, \forall f \in \mathcal{F}, p \in \mathcal{P}_f \quad (3d)$$

$$y_{f,p} \geq x_{f,p}, \ \forall f \in \mathcal{F}_{\text{Int}}, p \in \mathcal{P}_f \quad (3e)$$

$$u_{f,p} y_{f,p} - v_{f,p} x_{f,p} y_{f,p} \leq 0, \ \forall f \in \mathcal{F}_{\text{Int}}, p \in \mathcal{P}_f \quad (3f)$$

$$y_{f,p} \in \{0,1\}, \ \forall f \in \mathcal{F}_{\text{Int}}, p \in \mathcal{P}_f \quad (3g)$$

where (3b) ensures that no more bandwidth than required is allocated; (3c) is the link capacity constraint; and (3e)∼(3g) enforce the end-to-end delay constraint for interactive traffic.

MCTEQ is NP-hard. In MCTEQ, $y_{f,p}$ is a boolean variable. In fact, it is easy to verify that a continuous $y_{f,p}$ (i.e., $y_{f,p} \in [0,1]$) is still able to convey the end-to-end delay constraint through (3e) and (3f). The NP-hardness of MCTEQ comes from constraint (3f), where the bilinear term $x_{f,p}y_{f,p}$ ($0 \le x_{f,p} \le y_{f,p} \le 1$) makes the feasibility set of problem (3) not jointly convex in $\mathbf{x}$ and $\mathbf{y}$. The bilinear inequality constraint (3f) can be written as a bilinear matrix inequality (BMI). Any optimization problem over a BMI is NP-hard as shown in [10], since checking the resolvability of a BMI is NP-hard. Therefore, MCTEQ is also NP-hard.

C. Solving MCTEQ

We invoke the following two steps to transform the non-convex MCTEQ into a mixed integer program, where boolean variables exist only for interactive traffic and are not coupled with each other. In Sec. III, we use branch-and-bound method with linear programming (LP) to solve MCTEQ, which turns out to be very efficient.

Piecewise linear approximation of utility function. Since the inter-DC traffic management model needs to be run every short period (e.g., 5 minutes) – nearly on-line – response time is important. To this end, we use a piecewise linear approximation of the log utility function, i.e., making an LP approximation of the non-linear objective function (3a). This is done by: i) introducing an additional variable, ϕ_f, which corresponds to non-weighted utility achieved by FG f ; ii) adding a set of constraints, $\phi_f \le a_h \cdot (r_f x_f) + b_h, \forall f \in \mathcal{F}, h = 0, \cdots, H$, where H is the number of linear pieces used to approximate $\log(1+z), z \ge 0$. Now, ϕ_f replaces $\log(1 + r_f x_f)$. Hall et al. [11] provide a piecewise linear approximation to $\log_2(1+z)$ when z is in the range of $[0,1]$: $\log_2(1+z) \approx \min\{\tilde{a}_j z + \tilde{b}_j \mid 1 \le j \le N\}$. In our case, the maximum allocated bandwidth z for any FG f is assumed to be upper-bounded by a certain value B. Let $\Delta = \lceil \log_2 B \rceil$. We approximate $\log(1+z)$ in each segment of $[0, 2^0]$, $[2^0, 2^1]$, $\cdots$, $[2^{\Delta-1}, 2^\Delta]$ using N linear segments. That is, we need $H = N(\Delta+1)$ linear pieces in total to fully approximate $\log(1+z), 0 \le z \le B$. In segment $[2^{i-1}, 2^i]$, $\log_2(1+z) \approx \min\{\tilde{a}_j(\frac{1+z}{2^i} - 1) + i + \tilde{b}_j \mid 1 \le j \le N\}$, from which $(a_h, b_h), h = 0, \cdots, H$ can be derived. Fig. 2 illustrates the accuracy of the approximation for the case $N = 2$ and $B = 512$, leading to 20 linear segments being used to approximate $\log_2(1+z)$.

Linearization of constraint (3f). The bilinear term $x_{f,p}y_{f,p}$ ($x_{f,p} \in [0,1]$, $y_{f,p} \in \{0,1\}$) in (3f) can be linearized by introducing an extra variable $\xi_{f,p}$, that replaces $x_{f,p}y_{f,p}$ along with the following constraints: i) $\xi_{f,p} \le x_{f,p}$; ii) $\xi_{f,p} \le M y_{f,p}$ (M is a large positive factor); iii) $\xi_{f,p} \ge x_{f,p} - (1 - y_{f,p})M$; and iv) $\xi_{f,p} \ge 0$. Then, (3f) becomes $u_{f,p}y_{f,p} - v_{f,p}\xi_{f,p} \le 0$. Note that, after linearization, $y_{f,p}$ is still an integer variable and cannot be simply relaxed to a continuous variable.

D. Related Work and Alternative Solution Approaches

[1], [3] present a general SDN-based solution for inter-DC traffic management, while this paper focuses on proposing a faster and more efficient centralized traffic engineering algorithm. SWAN in [3] aims to achieve approximate max-min fairness of bandwidth allocation and deals with priorities of traffic classes different from MCTEQ. We compare MCTEQ with SWAN [3] in Sec. III. [12] also deals with the multi-class bandwidth allocation for inter-DC network by explicitly considering the throughput and delay requirements in the objective function, however, it differs from MCTEQ in characterizing the delay requirement. The semi-distributed algorithm derived in [12] is slow in convergence and is not SDN-friendly. We note that MCTEQ without the complicating delay constraints (3e)∼(3g), is similar to the traditional yet simpler multi-path flow control problem, which has been widely studied in the literature (see [13] and its references). Beside the delay constraints, the additional difficulty in MCTEQ lies in the requirement of a fast solution. The objective function (3a) is not strictly concave with respect to $x_{f,p}$, which implies the non-differentiability of the dual function. Many solutions have been proposed to deal with this issue. For example, [13] uses the proximal method that restores the strict concavity by adding a quadratic term to the objective function. However, all these solutions are proposed for distributed protocols and require a very large number of iterations to converge, incurring thus a high computational cost, making them hardly applicable to near-online traffic engineering and path reconfiguration. In Sec. II-C, we used approximation techniques to linearize (3a) and utilize commercial off-the-shelf efficient LP solvers to obtain very fast and efficient solution.

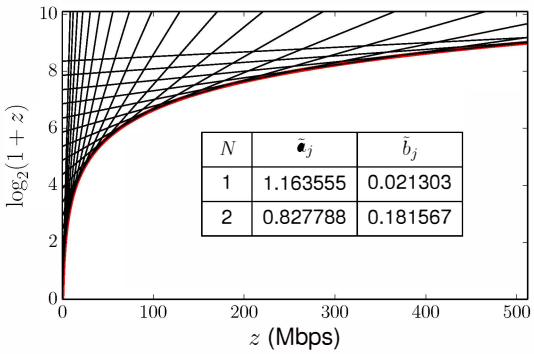


Fig. 2. Piecewise approximation of $\log_2(1+z)$ where $z \le 512$ and $N = 2$.

III. Evaluations

A. Setup and Methodology

G-WAN: Google's inter-DC backbone network comprises 12 DCs and 19 inter-DC links [1], as shown in Fig. 3. The link propagation delays are set based on the physical distances between cities where DCs are located (using the speed of light in optical fiber 2×10^8 m/s). We consider three settings of link capacities: i) **Type-A:** all links are 320 Gbps; ii) **Type-B:** each link is either 320 or 160 Gbps with equal probability; and iii) **Type-C:** the link capacities are estimated based on the gravity model and were rounded up to the nearest multiple of 80 Gbps to reflect common provisioning practices. Due to space constraints, we only report results for Type-A (homogeneous) and Type-C (heterogeneous) hereinafter.

Traffic demands: In our experiments, every DC pair has a demand in each traffic class. In total, the number of FGs is

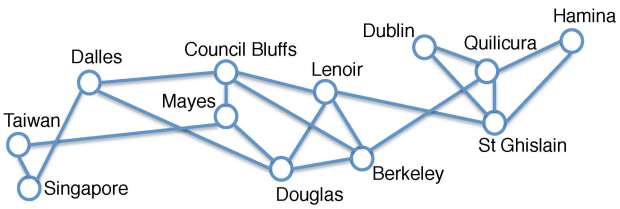


Fig. 3. G-WAN: the topology of Google's global data centers.

$12 \times 11 \times 3 = 396$. The demands of Interactive and Elastic are randomly generated. We scale their demands separately by a factor (via binary search) such that at least one link's load is nearly 99% under the multi-commodity flow model, as has been done in [3]. For Background traffic, demands are also randomly generated but have larger volumes than Interactive and Elastic traffic. The candidate tunnel sets $\{P_f\}$ for each FG are built by K-shortest paths between each pair of DCs, using Yen's algorithm [14]. The target delays ($\{\bar{d}_f\}$) for interactive traffic is set between [150, 200] ms. The token bucket buffer size σ_f is set to 10 MB.

Methodology: We compare four alternatives, summarized in Table II. The evaluation consists of five aspects. First, we compare the network utilization in terms of the total throughput of applications. Second, we observe the end-to-end delay bound for FGs in the Interactive class. Third, we observe the fairness of rate allocation for FGs in different classes. Fourth, we study the impact of the value of K (the size of P_f) on the total throughput. Finally, we report the computational time of the four alternatives.

TABLE II. FOUR INTER-DC BANDWIDTH ALLOCATION METHODS

Method	Explain
SWAN [3]	SWAN handles bandwidth allocation for FGs of different priorities in strict priority order; in each step, an approximate max-min fair allocation is achieved via a sequence of LPs.
MMF [15]	MMF achieves exact max-min fairness among FGs in the same class; we extend MMF to handle traffic priorities in the same way as SWAN, making SWAN an approximation to MMF.
MCTEQ	(3), which jointly allocates bandwidth for multi-classes with proportional fairness and enforces end-to-end delay constraints for interactive traffic.
MCTE	MCTE is simply MCTEQ where the end-to-end delay constraints (3e)∼(3g) are dropped.

B. Network Utilization

Table III reports how the four alternatives allocate bandwidth to the three traffic classes under Type-A & C link capacity settings. We make two observations. First, SWAN achieves similar results as MMF, while MCTEQ achieves similar results as MCTE. Second, MCTE and MCTEQ allocate a little less bandwidth to Interactive traffic, which enables them to allocate more bandwidth to Elastic and Background traffics. For example, MCTEQ yields 167 Gbps (under Type-A) and 159 Gbps (under Type-B) more throughput than SWAN. Therefore, MCTE and MCTEQ advocating joint bandwidth allocation for multi-class traffic can achieve a higher network utilization.

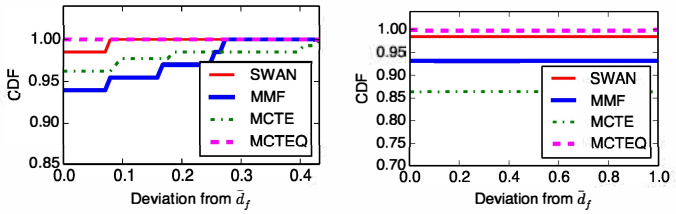
C. Delay of Interactive Flows

We calculate the maximum end-to-end delay bound, according to (2), for each FG based on the bandwidth allocation

TABLE III. BANDWIDTH ALLOCATION RESULTS OF FOUR METHODS.

	Link Capacity: Type-A				Link Capacity: Type-B			
Unit (**Gbps**)	Int	Ela	Bac	Sum	Int	Ela	Bac	Sum
SWAN	1272	638	1094	**3005**	640	401	867	**1908**
MMF	1272	696	996	2964	640	401	953	1995
MCTE	1253	713	1185	3151	631	409	1012	2052
MCTEQ	1234	732	1205	**3172**	613	409	1045	**2067**

results. Fig. 4 shows aggregated results. We make three observations. First, both MCTE and MMF cause a fraction (about 5%) of interactive FGs to violate their delay requirements. Second, SWAN modifies the objective function of max-min-fair allocation by adding another term [3], which favours tunnels with shorter propagation delays. We noticed that this small change did improve the results of MMF. However, SWAN cannot ensure that the target end-to-end delay is fully satisfied. For example, under Type-A setting, still 2% of FGs deviate from their target delays ($\bar{d}_f$) by 10%. Third, MCTEQ can always guarantee delay requirement as it takes this target explicitly into consideration in the problem formulation in contrast to MMF that does not consider it, and SWAN that employs a heuristic to improve the delay.



(a) LinkCap: Type-A (b) LinkCap: Type-C

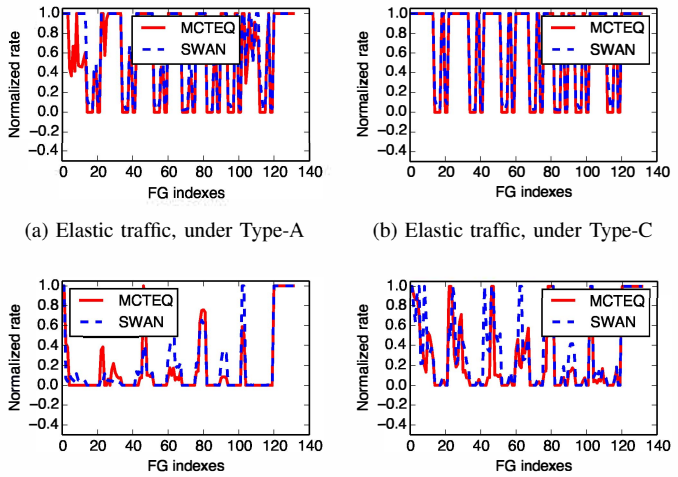
Fig. 4. End-to-end delay bound for FGs, when $K = 6$.

D. Fairness

Both SWAN and MCTEQ respect the priorities between traffic classes, albeit in a different manner. Interactive demands are almost fully satisfied (thus they are not shown here.) Fig. 5 shows the results of Elastic and Background traffic. The FG indices are in the increasing order of FG demands. The normalized rate is the ratio of the allocated bandwidth to the demand. We make two observations. First, a larger portion of demands of Elastic than Background are met. Second, SWAN and MCTEQ achieve similar fairness allocation in the same traffic classes.

E. Impact of K

The candidate sets of tunnel for FGs are constructed using K-shortest paths. Intuitively, to fully utilize the network's capacity, we need a larger K. Fig. 6 studies the trade-off between K and the achieved total throughput. We make three observations. First, MCTEQ achieves a higher throughput than SWAN consistently across different values of K. Second, $K = 6$ is already good enough to achieve a high network utilization. This is a nice property, as existing switch hardware limits the number of tunnels that can be installed. Third, further increasing K does not necessarily lead to higher throughput. In Fig. 6a, when K increases from 6, we see a clear decrease of SWAN's throughput. This is because having a larger

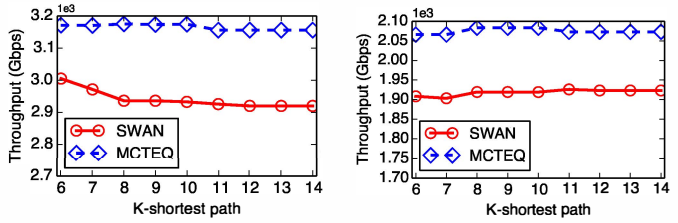


(a) Elastic traffic, under Type-A (b) Elastic traffic, under Type-C

(c) Background traffic, under Type-A (d) Background traffic, under Type-C

Fig. 5. The allocated bandwidth of Elastic and Background traffic classes.

K enables allocating more bandwidth to Interactive traffic, which yields more physical links to lack spare capacity and blocks further allocation to Elastic and Background traffic. Yet, MCTEQ does not have a clear decrease, due to the "joint allocation" approach.



(a) LinkCap: Type-A (b) LinkCap: Type-C

Fig. 6. The total network throughput achieved with varying K.

F. Computational Cost

We used Python2.7 to implement the algorithms and invoke Gurobi-5.6 [16] (with academic license) to solve the LP. We use a PC (running Windows 7) with Intel Core i3 CPU (3.30 GHz) and 4 GB RAM. Since the speed of completion of the allocation is critical to the on-line nature of inter-DC traffic management, we report the time (in seconds) it takes to run the four alternatives with different values of K in Table IV. From Table IV, we can see that both MCTE and MCTEQ are fast due to using the approximating technique in Sec. II-C to transform the problem in order to lean on the efficiency of the LP solver. SWAN is 2.5x faster than the classic MMF; MCTEQ is 2.5x faster than SWAN.

TABLE IV. RUNNING TIME (SECONDS) OF FOUR ALTERNATIVES

	K=6	K=8	K=9	K=10	K=11	K=12	K=14
SWAN	4.20	4.70	4.75	4.71	4.86	5.01	5.41
MMF	10.84	12.27	13.97	14.86	16.62	17.81	20.97
MCTE	1.38	1.21	1.30	1.36	1.42	1.57	1.71
MCTEQ	1.34	1.59	1.72	1.96	2.12	2.15	2.26

IV. Conclusion

We have formulated the MCTEQ problem to enable efficient inter-DC traffic management. MCTEQ advocates joint bandwidth allocation of multi-class traffic, leading to a higher network bandwidth utilization than alternative approaches such as SWAN that take steps to allocate bandwidth for classes in strict priority. In addition, MCTEQ can guarantee that the targeted end-to-end delay of Interactive flows is met, while alternative approaches such as SWAN can not promise this. Finally, to overcome the traditionally prohibitive computational complexity of the joint allocation, we proposed a new approximation to lean on the efficiency of LP solvers; as a result MCTEQ runs 2.5x faster the fastest alternative approach, which puts it in a very competitive position as a traffic engineering algorithm for SDN-based centralized intra-DC traffic management.

References

[1] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, J. Zolla, U. Hölzle, S. Stuart, and A. Vahdat, "B4: Experience with a globally-deployed software defined WAN," in *SIGCOMM'13*.

[2] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional ip routing protocols," *Communications Magazine, IEEE*, vol. 40, no. 10, pp. 118–124, Oct 2002.

[3] C.-Y. Hong, S. Kandula, R. Mahajan, M. Zhang, V. Gill, M. Nanduri, and R. Wattenhofer, "Achieving high utilization with software-driven WAN," in *SIGCOMM'13*.

[4] N. P. Katta, J. Rexford, and D. Walker, "Incremental consistent updates," in *HotSDN'13*.

[5] R. Mahajan and R. Wattenhofer, "On consistent updates in software defined networks," in *HotNets'13*.

[6] H. H. Liu, X. Wu, M. Zhang, L. Yuan, R. Wattenhofer, and D. Maltz, "zUpdate: Updating data center networks with zero loss," in *SIGCOMM'13*.

[7] A. Parekh and R. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the single-node case," *Networking, IEEE/ACM Transactions on*, vol. 1, no. 3, Jun 1993.

[8] P. Goyal, S. S. Lam, and H. M. Vin, "Determining end-to-end delay bounds in heterogeneous networks," *Multimedia Syst.*, vol. 5, no. 3, pp. 157–163, May 1997.

[9] W. Sun and K. G. Shin, "End-to-end delay bounds for traffic aggregates under guaranteed-rate scheduling algorithms," *IEEE/ACM Trans. Netw.*, vol. 13, no. 5, Oct. 2005.

[10] J. G. VanAntwerp and R. D. Braatz, "A tutorial on linear and bilinear matrix inequalities," *Journal of Process Control*, vol. 10, no. 4, 2000.

[11] E. L. Hall, D. D. Lynch, and S. J. Dwyer, "Generation of products and quotients using approximate binary logarithms for digital filtering applications," *IEEE Trans. Comput.*, vol. 19, no. 2, Feb. 1970.

[12] A. Ghosh, S. Ha, E. Crabbe, and J. Rexford, "Scalable multi-class traffic management in data center backbone networks," *Selected Areas in Communications, IEEE Journal on*, vol. 31, no. 12, 2013.

[13] X. Lin and N. Shroff, "Utility maximization for communication networks with multipath routing," *Automatic Control, IEEE Transactions on*, vol. 51, no. 5, pp. 766–781, May 2006.

[14] E. Martins and M. Pascoal, "A new implementation of yens ranking loopless paths algorithm," *Quarterly Journal of the Belgian, French and Italian Operations Research Societies*, vol. 1, no. 2, 2003.

[15] D. Nace and M. Pioro, "Max-min fairness and its applications to routing and load-balancing in communication networks: A tutorial," *Commun. Surveys Tuts.*, vol. 10, no. 4, Oct. 2008.

[16] I. Gurobi Optimization, "Gurobi optimizer reference manual," 2014. [Online]. Available: http://www.gurobi.com