Network Utility Maximization under Maximum Delay Constraints and Throughput Requirements

Qingyu Liu, Haibo Zeng, Minghua Chen

***Abstract*—We consider a multi-path routing problem of maximizing the aggregate user utility over a multi-hop network, subject to link capacity constraints, maximum end-to-end delay constraints, and user throughput requirements. A user's utility is a concave function of the achieved throughput or the experienced maximum delay. The problem is important for supporting real-time multimedia traffic and is uniquely challenging due to the need of simultaneously considering maximum delay constraints and throughput requirements. In this paper, we first show that it is NP-complete either (i) to construct a feasible solution strictly meeting all constraints, or (ii) to obtain an optimal solution after relaxing either the maximum delay constraints or the throughput requirements. We then develop a polynomial-time approximation algorithm named PASS. The design of PASS leverages a novel understanding between non-convex maximum-delay-aware problems and their convex average-delay-aware counterparts, which can be of independent interest and suggests a new avenue for solving maximum-delay-aware network optimization problems. We prove that PASS always obtains approximate solutions (i.e., with theoretical performance guarantees), at the cost of violating *both* the maximum delay constraints *and* the throughput requirements by up to *constant* ratios. We also develop two variants of PASS, named PASS-M and PASS-T, to generate approximate solutions at the cost of violating *either* the maximum delay constraints *or* the throughput requirements by up to *problem-dependent* ratios. We evaluate our solutions using extensive simulations on Amazon EC2 datacenters supporting video-conferencing traffic. Compared to the existing algorithms and a conceivable baseline, our solutions obtain up to 100% improvement of utilities, by meeting the throughput requirements but relaxing the maximum delay constraints to the extent acceptable for practical video conferencing applications.**

***Index Terms*—Delay-sensitive multiple-unicast network flow, delay-aware multi-path routing, network utility maximization**

I. Introduction

We consider a multiple-unicast communication scenario where there exist multiple network users, each of which streams a network flow from its source to its destination over a multi-hop network, possibly using multiple paths. We study the problem of maximizing the aggregate user utility, subject to link capacity constraints, maximum delay constraints, and user throughput requirements. A user's utility is a concave function of the achieved throughput or the experienced maximum delay. The maximum delay of a user denotes the maximum Source-to-Destination (S2D) delay, or equivalently the delay of the slowest S2D path that carries traffic.

Our study is motivated by the increasing interests of supporting delay-critical traffic in various applications, e.g., video conferencing [14]–[16]. It is reported that 51 million users per month attend WebEx meetings [17], and 3 billion minutes of calls per day use Skype [18]. Low S2D delay is vital for such video conferencing applications. As recommended by the International Telecommunication Union (ITU) [19], a delay of less than 150ms can provide transparent interactivity while delays above 400ms are unacceptable for video conferencing. We note that the maximum S2D delay, instead of the average one, is a critical concern for provisioning low delay services, since there may exist traffic which experiences an arbitrarily large S2D delay even for the solution that minimizes average S2D delay performance [10], [11]. In sharp contrast, all the traffic can be timely streamed from its source to its destination following any solution that has an acceptable maximum S2D delay performance, because the maximum S2D delay is defined as an upper bound of S2D delays of all traffic.

We consider a delay model where data transmission rate over a link is upper bounded by the link capacity, and data experiences a constant delay in traversing a link. End-to-end networking delay is known to be composed of processing delay, queuing delay, and propagation delay. Our constant delay model well captures the *traffic-independent* propagation delay, but does not consider the *traffic-dependent* processing delay or queuing delay. Although our constant delay model is special, our study under this model has both practical and theoretical significance, due to the following concerns:

(i) The constant delay model is suitable for a number of important real-world applications, particularly the routing of video conferencing traffic over inter-datacenter networks. According to recent reports from Google [20] and Microsoft [21], for most real-world inter-datacenter networks, cloud providers typically over-provision inter-datacenter link capacity by $2-3$ times on a dedicated backbone [20], and the average link-capacity utilization even for busy links is $30-60\%$ [21]. As such, most inter-datacenter flows can always be accommodated at their target rates [15]. The objective of flow assignment is thus to optimize many other critical performance metrics, e.g., network utility and delay, according to their throughput needs. For these inter-datacenter networks, link data transmission rate

Manuscript received July 5, 2019; revised January 7, 2020 and May 10, 2020; accepted June 14, 2020; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor I-Hong Hou. The work of Qingyu Liu and Haibo Zeng was supported in part by the NSF under Grant 1812963. The work of Minghua Chen was supported in part by a start-up grant at City University of Hong Kong, Project No. 9380118. A preliminary version of the work was presented at the ACM International Symposium on Mobile Ad Hoc Networking and Computing 2019 [1]. *(Corresponding authors: Qingyu Liu; Haibo Zeng; Minghua Chen.)*

Qingyu Liu and Haibo Zeng are with the Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24060 USA (e-mail: qyliu14@vt.edu; hbzeng@vt.edu).

Minghua Chen is with the School of Data Science, City University of Hong Kong, Hong Kong, China (e-mail: minghua.chen@cityu.edu.hk).

Digital Object Identifier

TABLE I
COMPARE OUR WORK WITH EXISTING STUDIES.

	Maximization Objective		Constraints		Networking Setting
	Aggregate Throughput-Based Utilities	Aggregate Maximum-Delay-Based Utilities	Throughput Requirements	Maximum Delay Constraints	Multiple-Unicast
Many, e.g., [2]–[5]	✓	✗	✓	✗	✓
[6]–[11]	✗	✓*	✓	✗	✗
[12], [13]	✓**	✗	✗	✓	✓
Our Work	✓	✓	✓	✓	✓

Note. *: The objective of [6]–[11] is to minimize maximum delay, which is a special case of maximizing maximum-delay-based utility functions.
**: The objective of [12], [13] is to maximize throughput, which is a special case of maximizing throughput-based utility functions.

cannot exceed link capacity, and the constant propagation delay dominates the delay for the data to traverse a link (it is practically justified by a real-world implementation on globally distributed Amazon EC2 datacenters in Section VI-A of [15]). These observations justify our delay model for the critical problem of routing video-conferencing traffic over real-world inter-datacenter networks.

(ii) The constant delay model is a first-step of modeling networks and helps obtain flows which do not congest links (link queuing delay remains negligible), by setting the link capacity lower than the link bandwidth. As the first study on network utility maximization with throughput requirements and maximum delay constraints, we assume a traffic-independent delay model to establish fundamental understandings of the problem. This delay model is not uncommon in the literature, e.g., it is used in [6], [7], [12], [13], [15]. Later in Section V, we generalize our results to the general traffic-dependent delay model and illustrate the challenges.

In this paper, we study a fundamental multiple-unicast network flow problem of maximizing the aggregate user utility subject to link capacity constraints, maximum delay constraints, and user throughput requirements. We summarize existing studies in Table I, and present detailed discussions in Section II. Briefly speaking, our study is the first work on the general network utility maximization problem under maximum delay constraints and user throughput requirements. In our problem, a user's utility is either a function of its achieved throughput or a function of its experienced maximum delay. For this general problem, we derive many fundamental results, which we believe can advance state-of-the-art and serve as benchmarks for future research in the area. Specifically, we make the following **contributions** for our problem.

B We prove that it is NP-complete either (i) to construct a feasible solution meeting all constraints, or (ii) to obtain an optimal solution after we relax maximum delay constraints or throughput requirements, due to the need of simultaneously considering maximum delay constraints and user throughput requirements. Thus, it is non-trivial to develop polynomial-time approximation algorithms even after we relax the maximum delay constraints or the throughput requirements.

B We design an algorithm named PASS (Polynomial-time Algorithm Supporting utility-maximal flows Subject to throughput/delay constraints). We leverage a novel understanding between non-convex maximum-delay-aware problems and their convex average-delay-aware counterparts, which suggests a new avenue for solving maximum-delay-aware network optimization problems. PASS obtains an approximate solution in polynomial time, after relaxing both maximum delay constraints and throughput requirements by up to constant ratios. (i) For the approximation ratio of PASS, we show that it is a constant for maximizing the throughput-based utilities, but it depends on the utility functions for maximizing the delay-based utilities; (ii) For the derived violation ratios of constraints of PASS, we show that there exist instances where solutions of PASS will violate constraints by ratios that are close to our derived ratios.

B By slightly modifying PASS, we design two other algorithms PASS-M and PASS-T. PASS-M (resp. PASS-T) obtains an approximate solution with a problem-dependent approximation ratio in polynomial time, after only relaxing throughput requirements (resp. maximum delay constraints) by up to problem-dependent ratios. We further prove that there does not exist either a constant ratio that can bound the violation of constraints, or a constant approximation ratio which can bound the performance gap as compared to the optimal, for PASS-M and PASS-T, for all problem instances in theory. Therefore, problem-dependent ratios are the best possible results for PASS-M and PASS-T.

B We evaluate the empirical performance of our algorithms in simulations of supporting video-conferencing traffic across Amazon EC2 datacenters. Compared to the existing algorithms as well as a conceivable baseline, our solutions can obtain up to 100% improvement of utilities, by meeting throughput requirements but relaxing maximum delay constraints to the extent acceptable for video conferencing applications.

II. RELATED WORK

There exist many network utility maximization studies with throughput concerns, e.g., [2]–[5], but only a few consider maximum delays. Since the maximum delay of a single-unicast flow is non-convex in the flow decision variables, even maximum-delay-aware problems under simple settings are usually NP-hard, e.g., the single-unicast maximum delay minimization problem, and challenging to solve [6].

Misra *et al.* [6] study the single-unicast maximum delay minimization problem subject to a throughput requirement, and design a Fully-Polynomial-Time Approximation Scheme (FPTAS). Zhang *et al.* [7] generalize the FPTAS of [6] and develop an FPTAS to minimize maximum delay subject to throughput, reliability, and differential delay constraints also in the single-unicast scenario. Both FPTASes require to solve flow problems iteratively in time-expanded networks, by

employing a binary-search based idea applicable only in the single-unicast setting. It is thus unclear how to extend their techniques to our general multiple-unicast scenario.

Cao *et al.* [12] develop an FPTAS that can maximize throughputs subject to maximum delay constraints in a multiple-unicast setting. This FPTAS is generalized by Yu *et al.* [13] to solve other throughput maximization problems. To satisfy maximum delay constraints while optimizing throughputs, FPTASes of [12], [13] require to solve flow problems iteratively in time-expanded networks, which is time-consuming. Moreover, the design of FPTASes in [12], [13] leverages the primal-dual algorithm, where their primal problems and associated dual problems need to be cast as linear programs. It is unclear how to extend their techniques to our general scenario where the utility of a unicast can be a concave function of the throughput.

We note that there exist other maximum-delay-aware studies in the literature. However, they only develop heuristic approaches. For example, Liu *et al.* [15] target the multicast maximum delay optimization problem. Their heuristic approach suffers from two limitations: (i) the running time could be high because the number of variables increases exponentially with the network size, and (ii) there is no theoretical performance guarantee of the achieved solution.

Overall, with the constant link delay model, existing maximum-delay-aware studies focus on either the throughput-constrained maximum delay minimization problem or the maximum-delay-constrained throughput maximization problem, which are just special cases of our problem. To design approximation algorithms, they rely on a technique of iteratively solving problems in expanded networks, leading to impractically high time complexities (e.g., at least $O(|E|^3|V|^4\mathcal{L})$ to minimize single-unicast maximum delay where $|V|$ is the number of nodes, $|E|$ is the number of links, and $\mathcal{L}$ is the input size of the given problem instance [6]). It is unclear how to generalize their techniques to our multiple-unicast utility maximization scenario, where the utility of a unicast is a concave function of the achieved throughput or the experienced maximum delay. In sharp contrast, we develop an approximation algorithm for our problem of maximizing utilities, by leveraging a novel understanding between non-convex maximum-delay-aware problems and their convex average-delay-aware counterparts, resulting in a small time complexity (e.g., $O(|E|^3\mathcal{L})$ to minimize single-unicast maximum delay in a dense network (Theorem 2)).

Instead of modeling link delay as a constant as in [6], [7], [12], [13], [15], there exist studies which model the link delay as a traffic-dependent function. For example, Correa *et al.* [8], [9] minimize maximum delay with delay-function-dependent approximation ratios. Liu *et al.* [10], [11] minimize maximum delay with constant approximation ratios. Our delay model is the same as those in [6], [7], [12], [13], [15] but different from [8]–[11]. We remark that maximum-delay-aware problems are fundamentally different with different delay models. For example, to minimize the single-unicast maximum delay, it is APX-hard (hence no PTAS exists unless P = NP) with the traffic-dependent delay model [9], but an FPTAS[1] exists with the constant delay model [6].

In the literature there also exist many studies which consider unreliable links where data transmission over a link only succeeds with a probability: (i) in a single-hop network, Hou *et al.* [23] propose scheduling policies for a set of sources to be feasible with respect to delay constraints and throughput requirements. In [24], Hou *et al.* extend their previous work and study the utility maximization problem. Deng *et al.* [25] further conduct a study on a similar problem but assuming a more general traffic pattern; (ii) in a multi-hop network, Hou [26] develops scheduling policies with delay and throughput taken into account. Singh and Kumar [27] study a similar problem of maximizing throughput subject to delay constraints. Those studies [23]–[27] are of little relevance with ours, because they assume the route is pre-determined, while we optimize route for maximizing network utility. Deng *et al.* [28] study a joint routing and scheduling problem which requires a small amount of link capacity redundancy to satisfy delay constraints and throughput requirements. Their focus is on designing online policies with good performance in terms of competitive ratio, which is very different from ours. Singh and Kumar [29] study a joint routing, scheduling, and power control problem of maximizing throughput under delay constraints, and in [30], Singh *et al.* consider a similar problem further subject to wireless link interference constraints. Studies [29], [30] both focus on designing distributed policies, and they leverage stochastic frameworks to take all randomness, e.g., the unreliability of links, into account. Hence they fundamentally differ from our study where no probabilistic information is involved.

III. Preliminary

A. System Model

We consider a multi-hop network modeled as a directed graph $G = (V, E)$ with $|V|$ nodes and $|E|$ links. Each link $e \in E$ has a constant capacity $c_e \geq 0$ and a constant delay $d_e \geq 0$. For each link $e \in E$, data streamed to e experiences a delay of d_e, and the rate of streaming data to e must be within the capacity c_e. We are given K users, where for each user i, a source $s_i \in V$ needs to stream a single-unicast network flow to a destination $t_i \in V \setminus \{s_i\}$, possibly using multiple paths.

We denote P_i as the set of simple paths[2] from s_i to t_i, and $P = \cup_{i=1}^{K} P_i$. For any $p \in P$, its path delay d^p is defined as

$$d^p = \sum_{e \in E : e \in p} d_e;$$

We denote a multiple-unicast network flow solution as $f = \{f_i; i = 1, 2, \ldots, K\}$, where a single-unicast flow f_i is defined as the assigned flow rate over P_i, i.e., $f_i = \{x^p : x^p \geq 0; p \in P_i\}$. For f_i, we define

$$x_i^e = \sum_{p \in P_i : e \in p} x^p$$

[1] Unless P = NP, it holds that FPTAS $\subsetneq$ PTAS in that the runtime of a PTAS is required to be polynomial in problem input but not $1/\epsilon$, while the runtime of an FPTAS is polynomial in both the problem input and $1/\epsilon$ [22].

[2] A simple path is a path which does not have repeating nodes.

as the aggregate link rate of $e \in E$ of the unicast i (or the user i equivalently). Similarly, we denote x_e as the total aggregate link rate of link $e \in E$, and

$$x_e \quad , \quad \sum_{i=1}^{K} x_i^e \quad = \quad \sum_{p2P:e2p} x^p:$$

We further denote the flow rate, or the **throughput** equivalently, achieved by a single-unicast flow f_i by $|f_i|$,

$$|f_i| \quad , \quad \sum_{p2P_i} x^p \quad = \quad \sum_{e2\mathsf{Out}(s_i)} x_i^e \quad = \quad \sum_{e2\mathsf{In}(t_i)} x_i^e;$$

where $\mathsf{Out}(v)$ (resp. $\mathsf{In}(v)$) is the set of outgoing (resp. incoming) links of v. The **maximum delay** experienced by f_i is defined as

$$\mathcal{M}(f_i) \quad , \quad \max_{p2P_i:x^p>0} d^p;$$

i.e., the delay of the longest (slowest) path with positive rates from s_i to t_i[3]. The total delay of f_i is

$$\mathcal{T}(f_i) \quad , \quad \sum_{p2P_i} (x^p \cdot d^p) \quad = \quad \sum_{e2E} (x_i^e \cdot d_e):$$

i.e., the summation of delays experienced by all flow units from s_i to t_i. With $\mathcal{T}(f_i)$, we can define the **average delay** experienced by f_i as $\mathcal{A}(f_i) \ , \ \mathcal{T}(f_i)=|f_i|$, i.e., the total delay normalized by the amount of flow. We let $\mathcal{A}(f_i) = 0$ if $|f_i| = 0$. Our definitions of throughput, maximum delay, and average delay are the same as those in related studies [6]–[10].

For each f_i, we denote its **throughput-based utility** as $\mathcal{U}_i^t(|f_i|)$, which is a function that rewards f_i based on the achieved throughput. We assume that $\mathcal{U}_i^t(|f_i|)$ is *concave, non-negative, and non-decreasing* with $|f_i| \geq 0$. Our assumptions on $\mathcal{U}_i^t(|f_i|)$ are realistic, as it is practically reasonable that the rate of increase in the throughput-based utility shall decrease with the throughput increasing rate, considering that larger the throughput is, more severely the network will be congested. As discussed in Section II, in the literature there exist many works of optimizing throughput-based network utility, e.g., [2]–[5], where their utility functions satisfy our assumptions.

For each f_i, we denote its **maximum-delay-based utility** as $-\mathcal{U}_i^d(\mathcal{M}(f_i))$, where the disutility $\mathcal{U}_i^d(\mathcal{M}(f_i))$ is a function that penalizes f_i based on the experienced maximum delay. We assume that $\mathcal{U}_i^d(\mathcal{M}(f_i))$ is *convex, non-negative, and non-decreasing* with $\mathcal{M}(f_i) \geq 0$. Our assumptions on $\mathcal{U}_i^d(\mathcal{M}(f_i))$ are realistic, as it is practically reasonable that the rate of increase in the delay-based disutility (the rate of decrease in the delay-based utility) shall increase with the delay increasing rate, considering that in real world our tolerance of communication delay becomes less as the delay becomes larger. To our best knowledge, we are the first to optimize delay-based network utility with the user's utility to be a general function of the experienced maximum delay.

[3]We call a path $p \in P_i$ with $x^p > 0$ as a flow-carrying path of f_i.

B. Problem Definition

We study the following problem of Maximizing aggregate user Utilities subject to link capacity constraints, maximum Delay constraints, and Throughput requirements (**MUDT**),

$$\begin{aligned}
(\mathsf{MUDT}): \text{obj:} \quad & \text{either } \max \sum_{i=1}^{K} \mathcal{U}_i^t(|f_i|); && (1a)\\
& \text{or } \max \ - \sum_{i=1}^{K} \mathcal{U}_i^d(\mathcal{M}(f_i)); && (1b)\\
\text{s.t.} \quad & |f_i| \ \geq \ R_i; \ \ \forall i = 1;2;:::;K; && (1c)\\
& \mathcal{M}(f_i) \ \leq \ D_i; \ \ \forall i = 1;2;:::;K; && (1d)\\
& f = \{f_1;f_2;:::;f_K\} \in \mathcal{X}; && (1e)
\end{aligned}$$

where $\mathcal{X}$ defines a feasible multiple-unicast flow f meeting flow conservation constraints and link capacity constraints, i.e.,

$$\begin{aligned}
\mathcal{X} \quad , \quad \Big(& \sum_{e2\mathsf{Out}(s_i)} x_i^e = \sum_{e2\mathsf{In}(t_i)} x_i^e = |f_i|; \ \forall 1 \leq i \leq K;\\
& \sum_{e2\mathsf{Out}(v)} x_i^e = \sum_{e2\mathsf{In}(v)} x_i^e; \ \forall v \in V \setminus \{s_i;t_i\}; \ \forall 1 \leq i \leq K;\\
& \sum_{i=1}^{K} x_i^e \leq c_e; \forall e \in E; \ \text{vars: } x_i^e \geq 0; \forall e; \forall i \quad : \Big)
\end{aligned}$$

In formula (1), the objective (1a) (resp. (1b)) maximizes the aggregate throughput-based utility (resp. maximum-delay-based utility), the throughput requirements (1c) ensure that the throughput achieved by each user i is no smaller than R_i, the maximum delay constraints (1d) restrict the maximum delay experienced by each user i to be no greater than D_i, and the feasibility constraint (1e) defines a feasible multiple-unicast network flow solution, meeting link capacity constraints.

C. A Generalization to Popular Communication Problems

MUDT is fundamentally critical as it generalizes several popular communication settings. Two representative settings are the Throughput-Constrained maximum Delay Minimization problem (TCDM) and the maximum-Delay-Constrained Utility Maximization (DCUM) problem.

TCDM aims to find a network flow to minimize the weighted summation of maximum delays of all users, subject to link capacity constraints and throughput requirements.

$$\begin{aligned}
(\mathsf{TCDM}): \min \quad & \sum_{i=1}^{K} (w_i \cdot \mathcal{M}(f_i)) && (2a)\\
\text{s.t.} \quad & |f_i| \geq R_i; \ \ \forall i = 1;2;:::;K; && (2b)\\
& f = \{f_1;f_2;:::;f_K\} \in \mathcal{X}; && (2c)
\end{aligned}$$

where in the objective (2a), a non-negative weight w_i is associated with the maximum delay of f_i for each $i = 1;2;:::;K$. TCDM is NP-hard, since as its special case when $K = 1$, the problem has been proven to be NP-hard [6]. Maximum delay minimization problems that are special cases of TCDM have been studied in [6], [8]–[10].

DCUM aims to find a network flow to maximize aggregate user utility, subject to link capacity constraints and maximum delay constraints. It has the following formulation.

$$(\text{DCUM}): \max \quad \sum_{i=1}^{K} \mathcal{U}_i^t(|f_i|) \tag{3a}$$
$$\text{s.t.} \quad \mathcal{M}(f_i) \le D_i; \quad \forall i = 1;2;:::;K; \tag{3b}$$
$$f = \{f_1; f_2;:::; f_K\} \in \mathcal{X}: \tag{3c}$$

DCUM is NP-hard, because as its special case when $K = 1$ and $\mathcal{U}_1^t(|f_1|) = |f_1|$, the problem can be proven to be NP-hard following a similar proof as introduced in the Appendix of [6]. As an example, [12] studies a throughput maximization problem that is a special case of DCUM.

By extending existing NP-hardness analysis on problems that are special cases of MUDT, in the following we give a theorem, which suggests that it is non-trivial to develop polynomial-time approximation algorithms for MUDT even subject to relaxed constraints.

Theorem 1: For MUDT, it is NP-complete (i) to construct a feasible solution that meets all constraints, or (ii) to obtain an optimal solution that meets throughput requirements but relaxes maximum delay constraints, or (iii) to obtain an optimal solution that meets maximum delay constraints but relaxes throughput requirements.

Proof: The proof is an easy adoption of Appendix of [6], and we refer it to Part A of supplementary materials. ∎

IV. Proposed Approximation Algorithms

We design an algorithm PASS to solve MUDT approximately in a polynomial time, at the cost of violating both throughput requirements and maximum delay constraints by constant ratios. We slightly modify PASS to get another two algorithms PASS-M and PASS-T, to obtain approximate solutions that can either strictly satisfy maximum delay constraints or strictly satisfy throughput requirements. Note again that in sharp contrast, existing maximum-delay-aware studies either minimize throughput-constrained maximum delay or maximize maximum-delay-constrained throughput, which are special cases of our problem MUDT. They rely on a time-consuming technique of solving problems iteratively in the time-expanded network to provide approximate solutions. Our PASS leverages a novel understanding between non-convex maximum-delay-aware problems and their convex average-delay-aware counterparts, which can be of independent interest and suggest a new avenue for solving maximum-delay-aware network optimization problems.

A. Algorithmic Structure of PASS

We note that the non-convex maximum delays bring difficulties for solving MUDT. The key idea of PASS is to replace the non-convex maximum delays in MUDT by the convex average delays, and solve the average-delay-aware counterpart instead. (i) We denote the average-delay-aware counterpart of the MUDT that maximizes throughput-based utilities, i.e., problem (1) with an objective of (1a), as **MUAT-T**. MUAT-T has the following formulation:

$$(\text{MUAT-T}): \text{obj:} \quad \max \quad \sum_{i=1}^{K} \mathcal{U}_i^t(|f_i|); \tag{4a}$$
$$\text{s.t.} \quad |f_i| \ge R_i; \ \forall i = 1;:::;K; \tag{4b}$$
$$\mathcal{T}(f_i) \le D_i \cdot |f_i|; \ \forall i = 1;:::;K; \tag{4c}$$
$$f = \{f_1; f_2;:::; f_K\} \in \mathcal{X}: \tag{4d}$$

(ii) We denote the average-delay-aware counterpart of the MUDT that maximizes maximum-delay-based utilities, i.e., problem (1) with an objective of (1b), as **MUAT-M**. MUAT-M has the following formulation:

$$(\text{MUAT-M}): \text{obj:} \quad \max \ -\sum_{i=1}^{K} \mathcal{U}_i^d(\mathcal{T}(f_i)=|f_i|); \tag{5a}$$
$$\text{s.t.} \quad |f_i| = R_i; \ \forall i = 1;:::;K; \tag{5b}$$
$$\mathcal{T}(f_i) \le D_i \cdot |f_i|; \ \forall i = 1;:::;K; \tag{5c}$$
$$f = \{f_1; f_2;:::; f_K\} \in \mathcal{X}: \tag{5d}$$

Note that in our formulation of MUAT-M, the throughput requirements (5b) are equality constraints. However, the throughput requirements (1c) of MUDT are inequality constraints. The motivation of using equality constraints instead of inequality ones in MUAT-M is as follows. If the throughput requirements are equality constraints, $|f_i|$ of each user i is a constant of R_i. This allows to replace the variable $|f_i|$ with the constant R_i, and makes the objective in (5a) a concave function of the variables. Otherwise, $|f_i|$ is a variable, and (5a) is no longer concave. In the following lemma, we prove that MUAT-M is the average-delay-aware counterpart of MUDT that maximizes maximum-delay-based utilities.

Lemma 1: MUAT-M is the average-delay-aware counterpart of MUDT that maximizes the maximum-delay-based utilities, in the sense that (i) it is the average-delay-aware counterpart of the following problem:

$$\text{obj:} \quad \max \ -\sum_{i=1}^{K} \mathcal{U}_i^d(\mathcal{M}(f_i)); \tag{6a}$$
$$\text{s.t.} \quad |f_i| = R_i; \ \forall i = 1;:::;K; \tag{6b}$$
$$\mathcal{M}(f_i) \le D_i; \ \forall i = 1;:::;K; \tag{6c}$$
$$f = \{f_1; f_2;:::; f_K\} \in \mathcal{X}; \tag{6d}$$

(ii) and the above problem formulated in (6) is equivalent to MUDT formulated in (1) with an objective of (1b).

Proof: Refer to Appendix VIII-A. ∎

Algorithm 1 presents PASS. It solves the average-delay-aware counterpart of MUDT and obtains the corresponding multiple-unicast flow solution $f = \{f_i; i = 1;:::;K\}$ (Line 5). Then for each $i = 1;:::;K$, we delete a rate of $\cdot|f_i|$ iteratively from the slowest flow-carrying paths of f_i (Line 8). In the end, the remaining flow is the solution of PASS.

B. Performance Guarantee of PASS

Lemma 2: In Algorithm 1 with an arbitrary $\in (0;1)$, suppose $\hat{f} = \{\hat{f}_i; i = 1;2;:::;K\}$ is the solution to the average-delay-aware counterpart of MUDT (Line 5), and

Algorithm 1 Our Proposed Algorithm PASS

1: **input**: Problem (1), $\in (0;1)$
2: **output**: $f = \{f_i; i = 1;2;:::;K\}$
3: **procedure**
4: Formulate either problem (4) or problem (5) that is the average-delay-aware counterpart of problem (1)
5: Solve the average-delay-aware problem and get the solution $f = \{f_i;\ i = 1;2;:::;K\}$
6: $x_i^{\text{delete}} = \cdot |f_i|;\ \forall i = 1;2;:::;K$
7: **for** $i = 1;2;:::;K$ **do**
8: **while** $x_i^{\text{delete}} > 0$ **do**
9: Find the slowest flow-carrying path $p_i \in P_i$
10: **if** $x^{p_i} > x_i^{\text{delete}}$ **then**
11: $x^{p_i} = x^{p_i} - x_i^{\text{delete}};\ x_i^{\text{delete}} = 0$
12: **else**
13: $x_i^{\text{delete}} = x_i^{\text{delete}} - x^{p_i};\ x^{p_i} = 0$
14: **return** the remaining flow $f = \{f_i; i = 1;2;:::;K\}$

$\bar{f} = \{\bar{f}_i; i = 1;2;:::;K\}$ is the solution returned in the end (Line 14). We have

$$\cdot \mathcal{M}(\bar{f}_i) \quad \leq \quad \mathcal{A}(\hat{f}_i);\quad \forall i = 1;2;:::;K: \tag{7}$$

Proof: Refer to Appendix VIII-B. ∎

Different from the proof in Appendix VIII-B, we remark that our Lemma 2 can be proved by Markov inequality as well. Lemma 2 suggests that the maximum delay of each single-unicast flow after deleting rate is bounded by a constant ratio as compared to the average delay of the corresponding single-unicast flow before deleting rate. This is a critical observation that theoretically relates the non-convex maximum delays with the convex average delays. In fact, by following our proof, it is easy to verify that Lemma 2 holds for any $\hat{f}$ and $\bar{f}$, as long as $\hat{f}$ is the flow before deleting -fraction rate from each single-unicast and $\bar{f}$ is the remaining flow after deleting -fraction rate from each single-unicast. The reason why PASS solves the average-delay-aware counterpart of MUDT to get the flow $\hat{f}$ is to provide the theoretical performance guarantee on the maximum delay constraint in a polynomial time.

Theorem 2: Given a feasible problem (1), suppose we use PASS (Algorithm 1) with an arbitrary $\in (0;1)$ to solve it. Then PASS must return a solution $\bar{f} = \{\bar{f}_i; i = 1;:::;K\}$ in polynomial time, meeting the following relaxed constraints

$$\bar{f}_i \quad \geq \quad (1 - \ ) \cdot R_i;\quad \forall i = 1;2;:::;K; \tag{8a}$$

$$\mathcal{M}\ \bar{f}_i \quad \leq \quad D_i = \ ;\quad \forall i = 1;2;:::;K; \tag{8b}$$

$$\bar{f} = \{\bar{f}_1; \bar{f}_2;:::;\bar{f}_K\} \in \mathcal{X}: \tag{8c}$$

Suppose $f^{\ } = \{f_i^{\ }; i = 1;2;:::;K\}$ is the optimal solution to problem (1). If the throughput-based utility maximization (1a) is the objective, $\bar{f}$ provides the following approximation ratio

$$\sum_{i=1}^{K} \mathcal{U}_i^t\ \bar{f}_i \quad \geq \ (1 - \ ) \cdot \sum_{i=1}^{K} \mathcal{U}_i^t\ (|f_i^{\ }|); \tag{9}$$

If the maximum-delay-based utility maximization (1b) is the objective, $\bar{f}$ provides the following approximation ratio

$$\sum_{i=1}^{K} \mathcal{U}_i^d\ \mathcal{M}\ \bar{f}_i \quad \leq \quad (\ ) \cdot \sum_{i=1}^{K} \mathcal{U}_i^d\ (\mathcal{M}\ (f_i^{\ }));\tag{10}$$

where $(\ )$ is defined as follows

$$(\ ) \ , \ \max_{i2f1;:::;Kg;\ 0<x\ \ D_i} \frac{\mathcal{U}_i^d(x = \ )}{\mathcal{U}_i^d(x)} \ :$$

Proof: Refer to Appendix VIII-C. ∎

It is clear that PASS obtains an approximate solution, at the cost of violating throughput requirements (1c) by a constant ratio of $(1 - \ )$, and violating maximum delay constraints (1d) by a constant ratio of $(1 = \ )$. If the objective is to maximize throughput-based utilities, the approximation ratio is $(1 - \ )$ which is a constant; otherwise if the objective is to maximize delay-based utilities, it is $(\ )$ which depends on the input delay-based utility functions. As an example, consider the n-order polynomial functions, i.e., $\mathcal{U}_i^d(\mathcal{M}(f_i)) = \sum_{j=0}^{n} c_{i;j} \cdot (\mathcal{M}(f_i))^j$ where $\{c_{i;j}; j = 0;1;:::;n\}$ are non-negative weights. We have $(\ ) = (1 = \ )^n$ for such polynomial utility functions, given any $\in (0;1)$ and any $D_i \geq 0$:

$$\begin{aligned}
\frac{\mathcal{U}_i^d(x = \ )}{\mathcal{U}_i^d(x)} \quad &= \quad \frac{\sum_{j=0}^{n} c_{i;j} \cdot (x = \ )^j}{\sum_{j=0}^{n} c_{i;j} \cdot x^j} \\
&= \quad \frac{1}{\ }^{n} \cdot \frac{\sum_{j=0}^{n} c_{i;j} \cdot x^j \cdot \ ^{n\ j}}{\sum_{j=0}^{n} c_{i;j} \cdot x^j} \\
&\leq \quad \frac{1}{\ }^{n} \cdot \frac{\sum_{j=0}^{n} c_{i;j} \cdot x^j}{\sum_{j=0}^{n} c_{i;j} \cdot x^j} \ = \ \frac{1}{\ }^{n} :
\end{aligned}$$

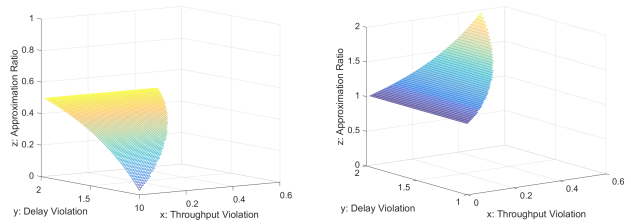
To obtain an approximate solution, according to Theorem 2, theoretically PASS needs to either violate delay constraints severely if throughput requirements are only allowed to be violated mildly, or violate throughput requirements severely if delay constraints are only allowed to be violated mildly. In fact, we remark that our derived ratios $(1 - \ )$ and $(1 = \ )$ of violating constraints have high quality and hence are useful for PASS. This is because they are constants independent of instances. Although they appear to be loose in some instances, in the following lemma we show that for any $\in (0;1)$, there always exists an instance where the solution of PASS violates constraints by ratios that are very close to them.

Lemma 3: Given any $\in (0;1)$, there exists an MUDT instance, where the following holds for the solution $\bar{f} = \{\bar{f}_i; i = 1;2;:::;K\}$ of PASS (Algorithm 1)

$$\begin{aligned}
\bar{f}_i \quad &\leq \ (1 - \ ) \cdot R_i;\quad \forall i = 1;2;:::;K; \\
\mathcal{M}\ \ \bar{f}_i \quad &\geq \ (\lceil 1 = \ \rceil - 1) \cdot D_i;\quad \forall i = 1;2;:::;K:
\end{aligned}$$

Proof: Refer to Part B of supplementary materials. ∎

We explain why there is a tradeoff between the ratio $(1 = \ )$ of violating maximum delay constraints and the ratio $(1 - \ )$ of violating throughput requirements over the next a few sentences. PASS resorts to solving the average-delay-aware counterpart to find a useful solution to the maximum-delay-aware problem MUDT. However, the proof of Lemma 3 suggests that there is an instance where the average-delay-optimal solution violates the maximum delay constraints severely, satisfying throughput requirements. This is because the delays of paths of the solution vary severely. To reduce the violation of maximum delay constraints, for this instance we need to delete a huge amount of flow rate from the average-delay-optimal solution, leading to a severe violation of throughput requirements.



(a) Approximation ratio for maximizing throughput-based utilities. (b) Approximation ratio for maximizing delay-based utilities.

Fig. 1. Approximation ratio of our PASS (z-axis), given a minimum violation ratio on the throughput requirements (x-axis) and a maximum violation ratio on the maximum delay constraints (y-axis).

Algorithm 2 PASS-M: Modify PASS to Strictly Meet Maximum Delay Constraints

1: **input**: Problem (1)
2: **output**: $f = \{f_i; i = 1; 2; :::; K\}$
3: **procedure**
4: Solve the average-delay-aware counterpart of problem (1), and get the solution $f = \{f_i; i = 1; 2; :::; K\}$
5: **for** $i = 1; 2; :::; K$ **do**
6: **while** $\mathcal{M}(f_i) > D_i$ **do**
7: Find the slowest flow-carrying path $p_i \in P_i$
8: Let $x^{p_i} = 0$
9: **return** the remaining flow $f = \{f_i; i = 1; 2; :::; K\}$

C. Applications of PASS

According to Theorem 2, we can control $\in (0; 1)$ to use PASS to obtain a solution with an approximation ratio of either $(1 -)$ or $( )$, at the cost of violating throughput requirements by a ratio of $(1 -)$ and violating maximum delay constraints by a ratio of $(1 =)$. Now we look at PASS from a different perspective. Instead of controlling an approximation parameter , suppose we can separately control a minimum violation ratio $x \in (0; 1)$ of throughput requirements and a maximum violation ratio $y \in (1; +\infty)$ of maximum delay constraints. We restrict that an acceptable solution f should satisfy the following:

$$|f_i| \geq x \cdot R_i; \ \mathcal{M}(f_i) \leq y \cdot D_i; \quad \forall i = 1; 2; :::; K: \tag{11}$$

We remark that we can use PASS to figure out such a solution: let us assume $^-$ to be the input approximation parameter of PASS. Based on Theorem 2, the following holds for the solution $\bar{f}$ of PASS:

$$\bar{f}_i \geq (1 - ^-) \cdot R_i; \ \mathcal{M} \ \bar{f}_i \leq D_i =^-; \quad \forall i = 1; 2; :::; K: \tag{12}$$

By comparing (12) with (11), it is clear if the following holds, $\bar{f}$ will satisfy the constraints in (11):

$$1 - ^- \geq x; \quad 1 =^- \leq y;$$

implying that $1 = y \leq ^- \leq 1 - x$. Therefore when $1 = y \leq 1 - x$, (i) PASS can figure out a solution meeting the constraints in (11), with an approximation ratio of $(1 - 1 = y)$ for maximizing throughput-based utilities, by setting $^- = 1 = y$; (ii) and PASS can figure out a solution meeting the constraints in (11), with an approximation ratio of $(1 - x)$ for maximizing delay-based utilities, by setting $^- = 1 - x$. If $1 = y > 1 - x$, PASS cannot obtain a solution to satisfy the constraints in (11). Considering an example with linear delay-based utilities, i.e., $\mathcal{U}_i^d(\mathcal{M}(f_i)) = w_i \cdot \mathcal{M}(f_i); i = 1; 2; :::; K$, we have $(1 - x) = 1 = (1 - x)$. We illustrate the approximation ratio of $(1 - 1 = y)$ (resp. $1 = (1 - x)$) of this example with the x and y in Figure 1(a) (resp. in Figure 1(b)).

For certain applications, the throughput requirements or the maximum delay constraints are hard constraints that cannot be violated. We note that one can use pre-scaled maximum delay constraints and throughput requirements as the input to PASS to generate feasible solutions as the output. Moreover, in the following, by slightly modifying PASS, we respectively develop (i) an algorithm PASS-M to achieve approximate solutions that can strictly meet maximum delay constraints, and (ii) an algorithm PASS-T to achieve approximate solutions that can strictly meet throughput requirements.

D. Modifying PASS to Meet Maximum Delay Constraints

We introduce PASS-M in Algorithm 2. Different from PASS that deletes $\cdot |f_i|$ rate from slowest flow-carrying paths of each f_i, PASS-M deletes rate from slowest flow-carrying paths of f_i till its maximum delay meets the constraint D_i.

Theorem 3: Given a feasible problem (1), suppose we use PASS-M (Algorithm 2) to solve it. Then PASS-M must return a solution $\bar{f} = \{\bar{f}_i; i = 1; 2; :::; K\}$ in polynomial time, meeting the following relaxed constraints

$$\bar{f}_i \geq (1 - _{\max}) \cdot R_i; \quad \forall i = 1; 2; :::; K; \tag{13a}$$

$$\mathcal{M} \ \bar{f}_i \leq D_i; \quad \forall i = 1; 2; :::; K; \tag{13b}$$

$$\bar{f} = \{\bar{f}_1; \bar{f}_2; :::; \bar{f}_K\} \in \mathcal{X}; \tag{13c}$$

where $_{\max}$ is defined as follows

$$_{\max} = \max_{1 \ i \ K} \left\{ \hat{f}_i - \bar{f}_i \ = \hat{f}_i \right\};$$

where $\hat{f} = \{\hat{f}_i; i = 1; 2; :::; K\}$ is the optimal solution to the average-delay-aware problem in Line 4 of Algorithm 2. Suppose $f = \{f_i ; i = 1; 2; :::; K\}$ is the optimal solution to problem (1). If the throughput-based utility maximization (1a) is the objective, $\bar{f}$ provides the following approximation ratio

$$\sum_{i=1} \mathcal{U}_i^t \ \bar{f}_i \geq (1 - _{\max}) \cdot \sum_{i=1} \mathcal{U}_i^t (|f_i|); \tag{14}$$

If the maximum-delay-based utility maximization (1b) is the objective, $\bar{f}$ provides the following approximation ratio

$$\sum_{i=1} \mathcal{U}_i^d \ \mathcal{M} \ \bar{f}_i \leq (_{\min}) \cdot \sum_{i=1} \mathcal{U}_i^d (\mathcal{M}(f_i)); \tag{15}$$

where $_{\min}$ is defined as follows

$$_{\min} = \min_{1 \ i \ K} \left\{ \hat{f}_i - \bar{f}_i \ = \hat{f}_i \right\};$$

Proof: It is a direct extension of Theorem 2. Detailed proof refers to Part C of supplementary materials. ■

Comparing Theorem 2 with Theorem 3, to solve MUDT, (i) PASS achieves an approximate solution at the cost of

violating both throughput requirements and maximum delay constraints by constant ratios, while (ii) PASS-M obtains an approximate solution and strictly meets maximum delay constraints, but at the cost of violating throughput requirements by a problem-dependent ratio. We highlight that although our derived problem-dependent ratios of PASS-M can be figured out only after we use PASS-M to solve MUDT, and can be arbitrarily bad for certain problem instances, they are the best effort for PASS-M as shown in the lemma below.

Lemma 4: Suppose $\bar{f} = \{\bar{f}_i; i = 1; 2; :::; K\}$ is the solution of PASS-M (Algorithm 2). Given any positive number that is arbitrarily close to 0, there exists an MUDT instance, where the following holds for $\bar{f}$:

$$\bar{f}_i \quad < \quad \cdot R_i; \quad \forall i = 1; 2; :::; K: \tag{16}$$

Suppose $f \quad = \{f_i ; i = 1; 2; :::; K\}$ is the optimal solution to MUDT. Given any positive number that is arbitrarily close to 0, there also exists an MUDT instance where the following holds for $\bar{f}$: If the throughput-based utility maximization is the objective, we have the following in this instance:

$$\sum_{i=1}^{K} \mathcal{U}_i^t \quad \bar{f}_i \quad < \quad \cdot \sum_{i=1}^{K} \mathcal{U}_i^t \left(|f_i|\right); \tag{17}$$

If the maximum-delay-based utility maximization is the objective, we have the following in this instance:

$$\sum_{i=1}^{K} \mathcal{U}_i^d \quad \mathcal{M} \quad \bar{f}_i \quad > \quad \frac{1}{-} \cdot \sum_{i=1}^{K} \mathcal{U}_i^d \left(\mathcal{M}\left(f_i\right)\right): \tag{18}$$

Proof: Refer to Part D of supplementary materials. ■

Lemma 4 suggests that there exist instances where the throughput of PASS-M is arbitrarily small and the utility of PASS-M is arbitrarily far from optimal. Therefore, we cannot derive a constant approximation ratio or a positive constant to bound the throughput requirements violation of PASS-M for all instances of MUDT. Our derived problem-dependent ratios are thus the best possible results for PASS-M.

E. Modifying PASS to Meet Throughput Requirements

In order to strictly meet throughput requirements, our PASS-T uses the optimal solution to the average-delay-aware counterpart of MUDT directly as a solution to the maximum-delay-aware problem MUDT, i.e.,

B PASS-T: directly solve the average-delay-aware counterpart of problem (1).

Theorem 4: Given a feasible problem (1), we denote $\bar{g} = \{\bar{g}_1; \bar{g}_2; :::; \bar{g}_K\}$ as the solution returned if we use PASS (Algorithm 1) to solve it with an $\in (0; 1)$. Now suppose we use PASS-T to solve problem (1). Then PASS-T must return a solution $\bar{f} = \{\bar{f}_i; i = 1; 2; :::; K\}$ in polynomial time, meeting the following relaxed constraints

$$\bar{f}_i \quad \geq \quad R_i; \quad \forall i = 1; 2; :::; K; \tag{19a}$$

$$\mathcal{M} \quad \bar{f}_i \quad \leq \quad - \cdot D_i; \quad \forall i = 1; 2; :::; K; \tag{19b}$$

$$\bar{f} = \{\bar{f}_1; \bar{f}_2; :::; \bar{f}_K\} \in \mathcal{X}; \tag{19c}$$

where is defined as follows

$$= \quad \max \quad 1; \max_{1 \quad i \quad K} \quad \mathcal{M}(\bar{f}_i) \text{=} \mathcal{M}(\bar{g}_i) \quad :$$

Suppose $f \quad = \{f_i ; i = 1; 2; :::; K\}$ is the optimal solution to problem (1). If the throughput-based utility maximization (1a) is the objective, $\bar{f}$ provides the following approximation ratio

$$\sum_{i=1}^{K} \mathcal{U}_i^t \quad \bar{f}_i \quad \geq \quad \sum_{i=1}^{K} \mathcal{U}_i^t \left(|f_i|\right); \tag{20}$$

If the maximum-delay-based utility maximization (1b) is the objective, $\bar{f}$ provides the following approximation ratio

$$\sum_{i=1}^{K} \mathcal{U}_i^d \quad \mathcal{M} \quad \bar{f}_i \quad \leq \quad ( ) \cdot \quad ( ) \cdot \sum_{i=1}^{K} \mathcal{U}_i^d \left(\mathcal{M}\left(f_i\right)\right): \tag{21}$$

Proof: It is a direct extension of Theorem 2. Detailed proof refers to Part E of supplementary materials. ■

Theorem 4 suggests that we can figure out an approximation ratio of PASS-T with the knowledge of an arbitrary solution of PASS. Comparing Theorem 2 with Theorem 4, to solve MUDT, (i) PASS achieves an approximate solution at the cost of violating both throughput requirements and maximum delay constraints by constant ratios, while (ii) PASS-T obtains an approximate solution and strictly meets throughput requirements, at the cost of violating maximum delay constraints by a problem-dependent ratio. Similar to PASS-M, although our derived problem-dependent ratios can be figured out only after we use PASS-T to solve MUDT, and can be unbounded for certain problem instances, they are the best effort for PASS-T as presented in the following lemma.

Lemma 5: Suppose $\bar{f} = \{\bar{f}_i; i = 1; 2; :::; K\}$ is the solution of PASS-T. Given an arbitrarily large number , there exists an MUDT instance, where the following holds for $\bar{f}$:

$$\mathcal{M} \quad \bar{f}_i \quad > \quad \cdot D_i; \quad \forall i = 1; 2; :::; K: \tag{22}$$

Suppose $f \quad = \{f_i ; i = 1; 2; :::; K\}$ is the optimal solution to MUDT. Given an arbitrarily large number , there also exists an MUDT instance where the following holds for $\bar{f}$: If the maximum-delay-based utility maximization is the objective, we have the following in this instance:

$$\sum_{i=1}^{K} \mathcal{U}_i^d \quad \mathcal{M} \quad \bar{f}_i \quad > \quad \cdot \sum_{i=1}^{K} \mathcal{U}_i^d \left(\mathcal{M}\left(f_i\right)\right): \tag{23}$$

Proof: Refer to Part F of supplementary materials. ■

Lemma 5 suggests that for PASS-T, we cannot derive a constant approximation ratio for maximizing the maximum-delay-based utility, or a positive constant to bound the maximum delay constraints violation, for all instances of MUDT. Therefore, our derived problem-dependent ratios are the best possible results for PASS-T.

F. Applicability to Other Maximum-Delay-Aware Problems

As shown in the formulation (1), MUDT has an objective of either (1a) or (1b), both of which maximize the aggregate user utility. We develop algorithms PASS, PASS-M, and PASS-T to solve MUDT approximately in previous sections. In this

section we consider two other user-utility-sensitive objectives which cover the objectives of MUDT as special cases. Our algorithms PASS, PASS-M, and PASS-T can optimize the general objectives under maximum delay constraints and user throughput requirements approximately, too.

A general extension of the function $\sum_{i=1}^{K} \mathcal{U}_i^t(|f_i|)$ is $\mathcal{U}^t(|f_1|;|f_2|;:::;|f_K|)$ that arbitrarily depends on the achieved throughput $|f_i|$ of each user $i; i = 1;2;:::;K$. Now we consider the following optimization objective:

$$\max \quad \mathcal{U}^t(|f_1|;|f_2|;:::;|f_K|); \tag{24}$$

where $\mathcal{U}^t(|f_1|;|f_2|;:::;|f_K|)$ is non-negative, non-decreasing, and concave with each $|f_i|; i = 1;2;:::;K$. Following the same proofs to Theorems 2, 3, and 4, it is easy to verify that we can use PASS, PASS-M, and PASS-T to approximately solve the problem with an objective of (24) subject to throughput requirements (1c), maximum delay constraints (1d), and feasibility constraints (1e), in polynomial time. A representative example of the general objective (24) which differs from the aggregate user utility maximization objective (1a) of MUDT is to maximize the worst utility among all users, i.e.,

$$\max \quad \min_{1 \le i \le K} \quad \mathcal{U}_i^t(|f_i|) \quad :$$

Similarly, we also consider the following delay-aware optimization objective that generalizes the aggregate user utility maximization objective (1b) of MUDT:

$$\max \quad -\mathcal{U}^d(\mathcal{M}(f_1);\mathcal{M}(f_2);:::;\mathcal{M}(f_K)); \tag{25}$$

where $\mathcal{U}^d(\mathcal{M}(f_1);\mathcal{M}(f_2);:::;\mathcal{M}(f_K))$ is non-negative, non-decreasing, and convex with each $\mathcal{M}(f_i); i = 1;2;:::;K$. It is easy to verify that we can use PASS, PASS-M, and PASS-T to approximately solve the problem with an objective of (25) subject to throughput requirements (1c), maximum delay constraints (1d), and feasibility constraints (1e), in polynomial time. Note that by optimizing the general objective (25), the approximation ratio $(\ )$ of PASS should be defined in the following way:

$$(\ ) \quad , \quad \max_{8i=1;2;:::;K:\ 0<x_i \quad D_i} \quad \frac{\mathcal{U}^d(x_1=\ ;x_2=\ ;:::;x_K=\ )}{\mathcal{U}^d(x_1;x_2;:::;x_K)} \quad :$$

A representative example of the general objective (25) is to maximize the worst utility among all users, i.e.,

$$\max \quad \min_{1 \le i \le K} \quad -\mathcal{U}_i^d(\mathcal{M}(f_i)) \quad :$$

Overall, we are the first to study the general network utility maximization problem under maximum delay constraints and throughput requirements, and propose algorithms with strong theoretical performance guarantees. The design of our algorithms further suggests a new avenue for solving a broad range of maximum-delay-aware network optimization problems. We believe that our fundamental results advance state-of-the-art, and can serve as benchmarks for future research.

V. Extension to Other Delay Models

In previous sections we study MUDT under a traffic-independent constant delay model. In this section we consider a traffic-dependent delay model where link delay is a function of link traffic. It covers the constant delay model as a special case. We highlight that when directly extending our algorithms from the traffic-independent delay model to the traffic-dependent one, (i) they maintain the same theoretical performance guarantee; however, (ii) their time complexities become exponential instead of polynomial.

Let us assume that the delay of link e is $d_e(x_e)$ which is a function of the link aggregate traffic x_e. It generalizes the constant link delay d_e in Section III. Due to practical concerns, we assume $d_e(x_e)$ to be non-negative, non-decreasing, differentiable, and convex with x_e. Now we focus on the MUDT under the traffic-dependent delay model.

Note that in our assumption $d_e(x_e)$ is non-decreasing with x_e. As presented in the last paragraph of Appendix VIII-B, clearly our Lemma 2 holds under the traffic-dependent delay model. Then following the same proof to Theorem 2 (resp. Theorem 3, Theorem 4), it is provable that the solution of PASS (resp. PASS-M, PASS-T) under the traffic-dependent delay model provides the same approximation ratio satisfying the same relaxed constraints, as compared to its performance guarantee under the constant delay model.

However, under the traffic-dependent delay model, the time complexities of PASS, PASS-M, and PASS-T are all exponential. This is because in such a model we have

$$\mathcal{T}(f_i) = \sum_{e \in E} (x_i^e \cdot d_e(x_e)) = \sum_{e \in E} x_i^e \cdot d_e\left(\sum_{i=1}^{K} x_i^e\right) \quad :$$

Here $\mathcal{T}(f_i)$ becomes non-convex with x_i^e. Therefore, the average-delay-aware counterpart of MUDT is a non-convex optimization problem. None of PASS, PASS-M, or PASS-T hence has a polynomial time complexity, as they all rely on solving the average-delay-aware counterpart to figure out useful solutions to MUDT.

Overall, our proposed avenue for maximum-delay-aware network optimization can be extended to the general setting of traffic-dependent delay, providing strong theoretical performance guarantee. This critical observation further highlights the theoretical significance of our results. Future directions include (i) developing efficient polynomial-time algorithms for MUDT under traffic-dependent delay model, by approximating the non-convex average-delay-aware counterpart of MUDT; and (ii) exploring real-world time-sensitive applications to which our proposed results can be directly applied.

VI. Performance Evaluation

Now we simulate a delay-critical video conferencing traffic over a real-world continent-scale inter-datacenter network topology of 6 globally distributed Amazon EC2 datacenters (see Figure 2). The network is modeled as a complete undirected graph. Each undirected link is treated as two directed links that operate independently and have identical delays and capacities, a common way to model an undirected graph by a directed one, e.g. in [31]. We set link delays and capacities

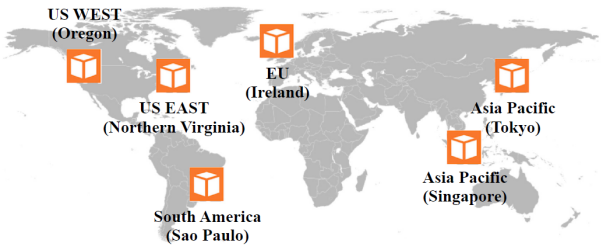
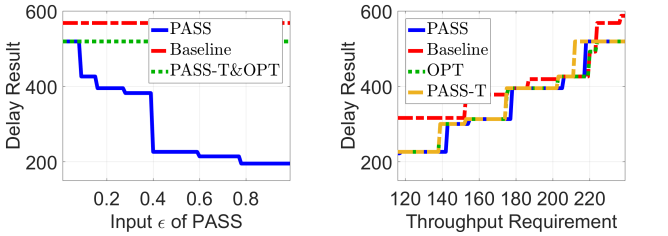


Fig. 2. Topology of the 6 Amazon EC2 datacenters [15].

TABLE II
INFORMATION OF $(d_e; c_e)$ IN THE AMAZON EC2 NETWORK [15], [16] (d_e IS IN MS AND c_e IS IN MBPS) (OR: OREGON, VA: VIRGINIA, IR: IRELAND, TO: TOKYO, SI: SINGAPORE, SP: SAO PAULO).

	OR	VA	IR	TO	SI	SP
OR	N/A	(41,82)	(86,86)	(68,138)	(117,74)	(104,67)
VA	-	N/A	(54,72)	(101,41)	(127,52)	(82,70)
IR	-	-	N/A	(138,56)	(117,44)	(120,61)
TO	-	-	-	N/A	(45,166)	(151,41)
SI	-	-	-	-	N/A	(182,33)
SP	-	-	-	-	-	N/A



(a) Delay results with of PASS, with $R_1 = R_2 = 230$.

(b) Delay results with throughput requirements, with = 3%.

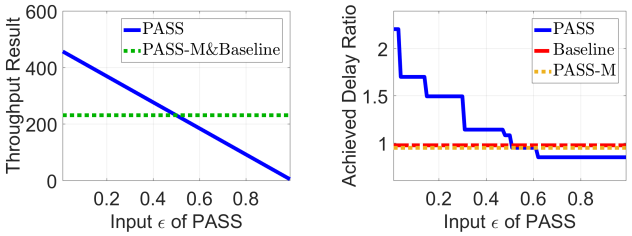
Fig. 3. Simulation results of using PASS to minimize the summation of maximum delays under throughput requirements.

according to practical evaluations on Amazon EC2 from [15], [16] (see Table II). We assume two unicasts ($K = 2$), one from Virginia to Singapore, the other from Oregon to Tokyo. Linear programs are solved using *CPLEX* [32].

A. Minimizing Maximum Delay

We first use our algorithms to minimize maximum delay, subject to link capacity constraints and throughput requirements (i.e., to solve TCDM with formula (2)). We assume $K = 2$, $w_1 = w_2$, and $R_1 = R_2 = R$ in (2).

We compare PASS with the optimal solution, a conceivable greedy baseline, and PASS-T respectively. Because link delays are all integers (see Table II), the delay of any path must be an integer. Therefore, we can obtain the optimal solution minimizing the summation of maximum delays, by enumerating all possible maximum delays of individual unicasts to figure out the minimal performance such that a feasible flow exists in the time-expanded network. Note that this approach theoretically has an exponential time complexity, and is the foundation of the FPTAS [6] designed for the single-unicast maximum delay minimization problem. The baseline greedily obtains the routing solution from the unicast 1 to the unicast K one by one. In the iteration of the unicast i, it assigns as much



(a) Throughput results (both baseline and PASS-M obtain the optimal).

(b) Delay ratio comparing the achieved result to the constraint.

Fig. 4. Simulation results of using PASS to maximize throughput under maximum delay constraints with various , where $D_1 = D_2 = 150$.

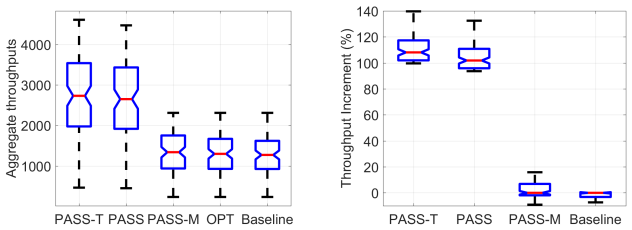
rate as possible to the shortest paths from s_i to t_i iteratively respecting the link capacity constraints, till the throughput requirement R_i is satisfied. Similar heuristic approaches have been used in other delay-aware network flow studies, e.g., in [33], yet without performance guarantee.

First, we evaluate the maximum delay of PASS with different values of (see Figure 3(a)). We set $R = 230$ and vary from 1% to 99% by a step of 1%. According to the figure, (i) PASS-T obtains the optimal solution to our problem, (ii) the delay of the baseline is strictly larger than optimal, and (iii) the delay of PASS is a staircase function with . We remark that the delay of PASS can be smaller than optimal in many instances because PASS only supports $(1 -$ $)$-fraction of the throughput requirement, while the optimal solution supports the full throughput requirement.

Second, we evaluate the maximum delay of PASS with the throughput requirement R (see Figure 3(b)). We set $= 3\%$ since a 3% throughput loss is very acceptable for video conferencing with protection/recovery capabilities [34]. We vary R from 116Mbps to 239Mbps with a step of 1Mbps. We remark that 116Mbps is the smallest throughput when the baseline needs multiple paths, and 239Mbps is the largest throughput that can be routed. Figure 3(b) suggests that PASS outputs a smaller maximum delay compared with the baseline in most instances. On average, the maximum delay of the baseline (402ms) is over 11% more than that of the optimal (362ms) and of the PASS (359ms). In the worst case ($R \in [116; 138]$), the maximum delay of the baseline is over 40% more than that of the optimal and of the PASS. In addition, PASS-T obtains the optimal solution to our problem in most instances, except for instances where $R \in [212; 223]$.

B. Maximizing Throughput

We then use our algorithms to maximize throughput, subject to link capacity constraints and maximum delay constraints (i.e., to solve DCUM with formula (3)). We assume $K = 2$, $\mathcal{U}_1^t(|f_1|) = |f_1|$, $\mathcal{U}_2^t(|f_2|) = |f_2|$, and $D_1 = D_2 = D$ in the formula (3). We compare PASS with the optimal solution, a conceivable baseline, and PASS-M, respectively. Similar to the greedy approach introduced in Section VI-A, the baseline assigns as much rate as possible to the shortest paths respecting both link capacity constraints and maximum delay constraints iteratively from unicast 1 to unicast 2 one by one. Besides,



(a) Throughput results of different algorithms.

(b) Throughput improvement results as compared to optimal.

Fig. 5. Simulation results of using PASS to maximize the weighted summation of throughputs subject to both maximum delay constraints and throughput requirements, with = 3%, $R_1 = R_2 = 80$ and $D_1 = D_2 = 150$.

similar to Section VI-A, we can obtain the optimal solution by solving problems in the time-expanded network.

We set $D = 150$ms due to the following concerns. (i) An end-to-end delay less than 150ms can provide a transparent interactivity for video conferencing [19]. (ii) A delay larger than 150ms (as long as it is less than 400ms) is still acceptable for video conferencing [19], and hence a solution that violates the maximum delay constraint may still be useful if it can achieve a substantial amount of throughput improvement.

We vary from 1% to 99% with a step of 1%. The throughput results are illustrated in Figure 4(a), and Figure 4(b) provides the achieved maximum delay ratios, i.e., $\max\{\mathcal{M}(f_1); \mathcal{M}(f_2)\}$=$D$ where $f = \{f_1; f_2\}$ is the solution. In our simulations, both the baseline and PASS-M obtain the optimal throughput while strictly meeting the maximum delay constraints. For $\leq 49\%$, the throughput of PASS is strictly larger than the optimal, while violating maximum delay constraints (e.g., 8% more than D when $= 49\%$). For $\geq 51\%$, the solution of PASS meets maximum delay constraints, but the achieved throughput is strictly smaller than optimal. It is impressive that with a small , e.g., $= 1\%$, the throughput of PASS is over 90% more than the optimal, while at the same time the maximum delays of PASS are less than 331ms which is still acceptable for video conferencing. For instances where $\leq 49\%$, when is decreased by 1%, on average a 2.0% throughput improvement as compared to the optimal can be achieved at the cost of a 2.2% violation to the maximum delay constraints.

C. Maximizing Network Utility

Finally we use PASS to maximize network utility subject to link capacity constraints, maximum delay constraints, and throughput requirements (i.e., to solve MUDT with formula (1)). We maximize the weighted summation of throughputs of individual users, i.e., $\mathcal{U}_i^t(|f_i|) = w_i \cdot |f_i|; i = 1; 2$, and we assume $R_1 = R_2 = 80$, $D_1 = D_2 = 150$.

We vary the weight w_1 (resp. w_2) from 1 to 10 with a step of 1, thus leading to 100 simulation instances each of which is characterized by a specific $\langle w_1; w_2 \rangle; 1 \leq w_1 \leq 10; 1 \leq w_2 \leq 10$. For each instance, we respectively run PASS, PASS-M, PASS-T, the conceivable baseline introduced in Section VI-B, and compare their solutions with the optimal. Note that we obtain the optimal solution by solving multiple-unicast flow problems in the time-expanded network.

We present the aggregate throughput results of different algorithms of the 100 simulation instances in Figure 5(a). In Figure 5(b), we give the throughput improvement of different algorithms as compared to the optimal. Note that PASS, PASS-M, and PASS-T can obtain utilities strictly greater than optimal, because they all optimize utility subject to relaxed constraints, while the optimal utility is achieved by a feasible solution strictly meeting all the constraints.

From Figure 5 we learn that PASS and PASS-T obtain a large improvement on the aggregate user throughput compared to the optimal (over 100% more than the optimal), while the aggregate user throughput achieved by PASS-M and the baseline is close-to-optimal. According to Theorem 2, theoretically PASS can violate both throughput requirements and maximum delay constraints. Empirically, (i) the throughput achieved by PASS is 138 (resp. 302) on average for the first unicast (resp. second unicast), both satisfying throughput requirements $R_1 = R_2 = 80$. (ii) The maximum delay experienced by PASS is 195 (resp. 301) on average for the first unicast (resp. second unicast), violating maximum delay constraints $D_1 = D_2 = 150$. But considering that video conferencing applications can accept a delay less than 400ms [19], the solution of PASS is acceptable. According to Theorem 3, theoretically PASS-M can meet maximum delay constraints while violate throughput requirements. Empirically, the throughput achieved by PASS-M is 71 (resp. 154) on average for the first unicast (resp. second unicast). It is clear that the first unicast flow violates throughput requirement. According to Theorem 4, theoretically PASS-T can meet throughput requirements while violate maximum delay constraints. Empirically, the maximum delay experienced by PASS-T is 222 (resp. 322) on average for the first unicast (resp. second unicast), violating the maximum delay constraints but within 400ms that is the largest acceptable delay.

VII. Conclusion

We consider the problem of maximizing aggregate user utilities subject to link capacity constraints, maximum delay constraints, and throughput requirements. A user's utility is a concave function of the achieved throughput or the experienced maximum delay. We first prove that it is NP-complete either (i) to construct a feasible solution meeting all constraints, or (ii) to obtain an optimal solution after we relax maximum delay constraints or throughput requirements. We then design the first polynomial-time approximation algorithm named PASS to obtain an approximate solution, at the cost of violating both maximum delay constraints and throughput requirements by up to constant ratios. By slightly modifying PASS, we develop two algorithms PASS-M and PASS-T to obtain approximate solutions at the cost of violating either maximum delay constraints or throughput requirements by up to problem-dependent ratios. Our results can serve as benchmarks for future research in the area. The design of our algorithms leverages a new understanding between maximum-delay-aware problems and their average-delay-aware counter-

parts. It suggests a new avenue for solving a broad range of maximum-delay-aware network optimization problems.

REFERENCES

[1] Q. Liu, H. Zeng, and M. Chen, "Network utility maximization under maximum delay constraints and throughput requirements," in *Proc. ACM Int'l Sym. Mobile Ad Hoc Networking and Computing*, 2019, pp. 391–392.

[2] F. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research society*, vol. 49, no. 3, pp. 237–252, 1998.

[3] S. H. Low and D. E. Lapsley, "Optimization flow control—i: basic algorithm and convergence," *IEEE/ACM Trans. Networking*, vol. 7, no. 6, pp. 861–874, 1999.

[4] J. Wang, L. Li, S. H. Low, and J. C. Doyle, "Can shortest-path routing and tcp maximize utility," in *Proc. IEEE Int'l Conf. Computer Communications*, 2003, pp. 2049–2056.

[5] D. P. Palomar and M. Chiang, "A tutorial on decomposition methods for network utility maximization," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1439–1451, 2006.

[6] S. Misra, G. Xue, and D. Yang, "Polynomial time approximations for multi-path routing with bandwidth and delay constraints," in *Proc. IEEE Int'l Conf. Computer Communications*, 2009, pp. 558–566.

[7] W. Zhang, J. Tang, C. Wang, and S. de Soysa, "Reliable adaptive multi-path provisioning with bandwidth and differential delay constraints," in *Proc. IEEE Int'l Conf. Computer Communications*, 2010, pp. 1–9.

[8] J. R. Correa, A. S. Schulz, and N. E. S. Moses, "Computational complexity, fairness, and the price of anarchy of the maximum latency problem," in *Proc. Int'l Conf. Integer Programming and Combinatorial Optimization*, 2004, pp. 59–73.

[9] J. Correa, A. Schulz, and N. Stier-Moses, "Fast, fair, and efficient flows in networks," *Operations Research*, vol. 55, no. 2, pp. 215–225, 2007.

[10] Q. Liu, L. Deng, H. Zeng, and M. Chen, "A tale of two metrics in network delay optimization," in *Proc. IEEE Int'l Conf. Computer Communications*, 2018, pp. 2123–2131.

[11] ——, "A tale of two metrics in network delay optimization," *IEEE/ACM Trans. Networking*, vol. 28, no. 3, pp. 1241–1254, 2020.

[12] Z. Cao, P. Claisse, R.-J. Essiambre, M. Kodialam, and T. Lakshman, "Optimizing throughput in optical networks: The joint routing and power control problem," *IEEE/ACM Trans. Networking*, vol. 25, no. 1, pp. 199–209, 2017.

[13] R. Yu, G. Xue, and X. Zhang, "Application provisioning in fog computing-enabled internet-of-things: A network perspective," in *Proc. IEEE Int'l Conf. Computer Communications*, 2018, pp. 783–791.

[14] X. Chen, M. Chen, B. Li, Y. Zhao, Y. Wu, and J. Li, "Celerity: a low-delay multi-party conferencing solution," in *Proc. ACM Int'l Conf. Multimedia*, 2011, pp. 493–502.

[15] Y. Liu, D. Niu, and B. Li, "Delay-optimized video traffic routing in software-defined interdatacenter networks," *IEEE Trans. Multimedia*, vol. 18, no. 5, pp. 865–878, 2016.

[16] M. Hajiesmaili, L. T. Mak, Z. Wang, C. Wu, M. Chen, and A. Khonsari, "Cost-effective low-delay design for multiparty cloud video conferencing," *IEEE Trans. Multimedia*, vol. 19, no. 12, pp. 2760–2774, 2017.

[17] WebEx, 2017. [Online]. Available: https://blog.webex.com/2016/01/five-reasons-to-join-a-webex-now/

[18] Skype, 2017. [Online]. Available: https://news.microsoft.com/bythenumbers/skype-calls

[19] ITU, "Series g: Transmission systems and media, digital systems and networks," International Telecommunication Union, Geneva, Switzerland, 2003.

[20] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu *et al.*, "B4: Experience with a globally-deployed software defined wan," in *Proc. ACM SIGCOMM Computer Communication Review*, 2013, pp. 3–14.

[21] C.-Y. Hong, S. Kandula, R. Mahajan, M. Zhang, V. Gill, M. Nanduri, and R. Wattenhofer, "Achieving high utilization with software-driven wan," in *Proc. ACM SIGCOMM Computer Communication Review*, 2013, pp. 15–26.

[22] WIKI. (2017) Polynomial time approximation scheme. [Online]. Available: https://en.wikipedia.org/wiki/Polynomial-time_approximation_scheme

[23] I.-H. Hou, V. Borkar, and P. Kumar, "A theory of qos for wireless," in *Proc. IEEE Int'l Conf. Computer Communications*, 2009.

[24] I.-H. Hou and P. Kumar, "Utility maximization for delay constrained qos in wireless," in *Proc. IEEE Int'l Conf. Computer Communications*, 2010.

[25] L. Deng, C.-C. Wang, M. Chen, and S. Zhao, "Timely wireless flows with general traffic patterns: Capacity region and scheduling algorithms," *IEEE/ACM Trans. Networking*, vol. 25, no. 6, pp. 3473–3486, 2017.

[26] I.-H. Hou, "Packet scheduling for real-time surveillance in multihop wireless sensor networks with lossy channels," *IEEE Trans. Wireless Communications*, vol. 14, no. 2, pp. 1071–1079, 2014.

[27] R. Singh and P. R. Kumar, "Decentralized throughput maximizing policies for deadline-constrained wireless networks," in *Proc. IEEE Conf. Decision and Control*, 2015, pp. 3759–3766.

[28] H. Deng, T. Zhao, and I.-H. Hou, "Online routing and scheduling with capacity redundancy for timely delivery guarantees in multihop networks," *IEEE/ACM Trans. Networking*, vol. 27, no. 3, pp. 1258–1271, 2019.

[29] R. Singh and P. Kumar, "Throughput optimal decentralized scheduling of multihop networks with end-to-end deadline constraints: Unreliable links," *IEEE Trans. Automatic Control*, vol. 64, no. 1, pp. 127–142, 2018.

[30] R. Singh, P. Kumar, and E. Modiano, "Throughput optimal decentralized scheduling of multi-hop networks with end-to-end deadline constraints: Ii wireless networks with interference," *arXiv preprint arXiv:1709.01672*, 2017.

[31] B. Grimmer and S. Kapoor, "Nash equilibrium and the price of anarchy in priority based network routing," in *Proc. IEEE Int'l Conf. Computer Communications*, 2016, pp. 1–9.

[32] IBM, "Cplex optimizer," 2017. [Online]. Available: https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/

[33] F. Devetak, J. Shin, T. Anjali, and S. Kapoor, "Minimizing path delay in multipath networks," in *Proc. IEEE Int'l Conf. Communications*, 2011, pp. 1–5.

[34] I. M. Weinstein, "Polycom's lost packet recovery (lpr) capability," *Wainhouse Research*, 2008. [Online]. Available: http://docs.polycom.com/global/documents/whitepapers/lost_packet_recovery_eval_report.pdf

[35] F. Potra and Y. Ye, "A quadratically convergent polynomial algorithm for solving entropy optimization problems," *SIAM Journal on Optimization*, vol. 3, no. 4, pp. 843–860, 1993.

[36] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric algorithms and combinatorial optimization*. Springer Science & Business Media, 2012.

[37] Y. Ye, "An $O(n^3L)$ potential reduction algorithm for linear programming," *Mathematical programming*, vol. 50, no. 1-3, pp. 239–258, 1991.

[38] L. R. Ford and D. R. Fulkerson, "Maximal flow through a network," *Canadian journal of Mathematics*, vol. 8, no. 3, pp. 399–404, 1956.

VIII. APPENDIX

A. Proof to Lemma 1

Proof: We prove that (6) is feasible if and only if (1) with an objective of (1b) is feasible, and they share the same optimal solution.

Only if part. Suppose the problem formulated in (6) is feasible. Because any feasible solution to the problem formulated in (6) must also be feasible to MUDT formulated in (1) with an objective of (1b), it holds that MUDT formulated in (1) with an objective of (1b) must be feasible.

If part. Suppose MUDT formulated in (1) with an objective of (1b) is feasible, and $f = \{f_1; ...; f_K\}$ is an arbitrary feasible solution to it. For each unicast $i = \{1; ...; K\}$, if $|f_i| > R_i$, we delete flow rate from f_i till $|f_i| = R_i$; otherwise we do nothing. Then we can always obtain a solution $g = \{g_1; ...; g_K\}$ from f such that $|g_i| = R_i$ for all i, meeting constraints (6b) and (6d). It is clear that $\mathcal{M}(g_i) \leq \mathcal{M}(f_i)$ because link delay is a constant, which implies that g satisfies (6c). Thus g is feasible to the problem formulated in (6), implying that the problem formulated in (6) is theoretically feasible.

We prove that the two problems share the same optimal solution by contradiction. Suppose $g = \{g_i; i = 1; 2; ...; K\}$ is the optimal solution to the problem formulated in (6),

$f = \{f_i; i = 1; 2; :::; K\}$ is the optimal solution to MUDT formulated in (1) with an objective of (1b), and

$$\sum_{i=1}^{K} \mathcal{U}_i^d(\mathcal{M}(g_i)) > \sum_{i=1}^{K} \mathcal{U}_i^d(\mathcal{M}(f_i)):$$

Note that $\sum_{i=1}^{K} \mathcal{U}_i^d \mathcal{M}(g_i) < \sum_{i=1}^{K} \mathcal{U}_i^d(\mathcal{M}(f_i))$ does not holds because g is feasible to MUDT.

As introduced in the previous proof, we can construct a g from f where g is feasible to the problem formulated in (6), and $\mathcal{M}(g_i) \le \mathcal{M}(f_i); \forall i = 1; 2; :::; K$. Considering that utility functions are non-decreasing, we have

$$\sum_{i=1}^{K} \mathcal{U}_i^d(\mathcal{M}(f_i)) \ge \sum_{i=1}^{K} \mathcal{U}_i^d(\mathcal{M}(g_i));$$

implying that

$$\sum_{i=1}^{K} \mathcal{U}_i^d(\mathcal{M}(g_i)) > \sum_{i=1}^{K} \mathcal{U}_i^d(\mathcal{M}(g_i));$$

which is contradicted with that g is optimal to (6). Therefore, the two problems share the same optimal solution.

After we replace the maximum delays in (6) by the average delays, clearly we get MUAT-M formulated in (5). Because we prove that (6) is feasible if and only if (1) with an objective of (1b) is feasible, and they share the same optimal solution, it holds that MUAT-M is the average-delay-aware counterpart of MUDT that maximizes delay-based utilities. ■

B. Proof to Lemma 2

Proof: According to Algorithm 1, for any $i = 1; 2; :::; K$, $\bar{f}_i$ is obtained by iteratively deleting $\cdot |\hat{f}_i|$ rate from $\hat{f}_i$. Suppose that there are in total N_i iterations to get $\bar{f}_i$ by deleting rate from $\hat{f}_i$ (namely assume N_i to be the number of iterations of the while-loop of line 8). And we use f_i^n to represent the flow of the unicast i at the beginning of the n-th iteration (or equivalently, at the end of the $(n-1)$-th iteration). Obviously, $f_i^1 = \hat{f}_i$, $f_i^{N_i+1} = \bar{f}_i$. We denote P_i^n as the set of of all *flow-carrying* paths in flow f_i^n, and $p_i^n \in P_i^n$ as the slowest flow-carrying path in P_i^n. In the n-th iteration of the unicast i, PASS delete some rate, say $x_i^n > 0$, from p_i^n.

Since all link delays are non-negative constants, the path delay cannot increase with reduced flow rate. Thus,

$$\mathcal{M}(f_i^{n+1}) \le \mathcal{M}(f_i^n); \quad \forall n = 1; 2; :::; N_i; \forall i = 1; 2; :::; K: \tag{26}$$

For any $1 \le n \le N_i$, the following held for any i

$$\begin{aligned}
\mathcal{T}(f_i^n) &= \sum_{e2E:e2p_i^n} [x_i^e d_e] + \sum_{e2E:e2p_i^n} [x_i^e d_e] \\
&= \sum_{e2E:e2p_i^n} [x_i^e d_e] + \sum_{e2E:e2p_i^n} [(x_i^e - x_i^n) d_e + x_i^n d_e] \\
&\overset{(a)}{=} \sum_{e2E:e2p_i^n} [x_i^e d_e] + \sum_{e2E:e2p_i^n} [(x_i^e - x_i^n) d_e] + x_i^n \mathcal{M}(f_i^n) \\
&\overset{(b)}{=} \mathcal{T}(f_i^{n+1}) + x_i^n \mathcal{M}(f_i^n) \overset{(c)}{\ge} \mathcal{T}(f_i^{n+1}) + x_i^n \mathcal{M}(\bar{f}_i):
\end{aligned} \tag{27}$$

In (27), equality (a) holds because $\sum_{e2p_i^n} d_e$ is the path delay of the slowest flow-carrying path p_i^n. Equality (b) holds because flow f_i^{n+1} is the flow when f_i^n deletes x_i^n rate from path p_i^n. Inequality (c) comes from (26) and $f_i^{N_i+1} = \bar{f}_i$.

We then do summation for (27) over $n \in [1; N_i]$, and get

$$\begin{aligned}
\mathcal{T}(\hat{f}_i) &= \mathcal{T}(f_i^1) \ge \mathcal{T}(f_i^{N_i+1}) + \sum_{n=1}^{N_i} x_i^n \cdot \mathcal{M}(\bar{f}_i) \\
&= \mathcal{T}(\bar{f}_i) + \cdot \hat{f}_i \cdot \mathcal{M}(\bar{f}_i);
\end{aligned}$$

which proves our Lemma 2 since it holds that $\mathcal{T}(\bar{f}_i) \ge 0$.

Finally, note that our constant delay model is sufficient but not necessary for our Lemma 2 to hold. Following the similar proof, it is easy to verify that our Lemma 2 holds if for each link we have that the link delay does not increase when the flow rate assigned to the link decrease. ■

C. Proof to Theorem 2

Proof: First, we prove the polynomial time complexity. Both problem (4) and (5) can be solved in polynomial time, since (i) they are convex programs with a polynomial size, and (ii) convex programming problems can be solved up to an arbitrarily small additive error in polynomial time (e.g., see [35], [36] for details). For example, the time complexity is $O(|E|^3 K^3 \mathcal{L})$ where $\mathcal{L}$ is the input size of the instance of the problem (4) or (5) if they are linear programs [37].

After solving the average-delay-aware problem, we get K single-unicast flows each of which is defined on edges. By the classic flow decomposition technique [38], we can then achieve K single-unicast flows $\hat{f} = \{\hat{f}_i; i = 1; 2; :::; K\}$ each of which is defined on paths within a time of $O(|V|^2|E|K)$. Note that the flow decomposition outputs at most $|E|$ paths for each $\hat{f}_i$, and hence there are at most $|E|$ iterations to obtain each $\bar{f}_i$ by deleting rate from $\hat{f}_i$. Overall, Algorithm 1 has a polynomial time complexity that is even independent to .

Second, we prove the existence of $\bar{f}$.

(i) Suppose (1a) is the objective of the problem (1). Because problem (1) is feasible and f is its optimal solution, f must satisfy all the constraints of problem (1), implying that f also satisfies the constraints (4b) and (4d) of the problem (4) that is the average-delay-aware counterpart of the problem (1). Now consider that we have $\mathcal{T}(g) \le \mathcal{M}(g) \cdot |g|$ for any single-unicast flow g, for any $i = 1; 2; :::; K$, the following holds

$$\mathcal{T}(f_i) \le \mathcal{M}(f_i) \cdot |f_i| \overset{(a)}{\le} D_i \cdot |f_i|;$$

where the inequality (a) comes from that f meets the constraints (1d). Therefore, f is also a feasible solution to the problem (4). Due to the existence of f , (4) must be feasible and hence Algorithm 1 must return a solution $\bar{f}$.

(ii) Suppose (1b) is the objective of the problem (1). Because problem (1) is feasible and f is its optimal solution, f must meet all the constraints of problem (1), e.g., we have $|f_i| \ge R_i; \forall i = 1; 2; :::; K$. Now we construct another network flow f based on f as follows: for each $i = 1; 2; :::; K$, we obtain f_i directly from f_i, by deleting flow rate from arbitrary flow-carrying paths of f_i till $|f_i| = R_i$. The