Cost Minimization in Multi-Path Communication under Throughput and Maximum Delay Constraints

Qingyu Liu*, Haibo Zeng*, Minghua Chen†, Lingjia Liu*
*Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, USA
†School of Data Science, The City University of Hong Kong, Hong Kong, China

***Abstract*—We consider the scenario where a sender streams a flow at a fixed rate to a receiver across a multi-hop network, possibly using multiple paths. Data transmission over a link incurs a cost and a delay, both of which are traffic-dependent. We study the problem of minimizing network transmission cost subject to a maximum delay constraint and a throughput requirement. The problem is important for leveraging edge-cloud computing platforms to support computationally intensive IoT applications, which are sensitive to three critical performance metrics, i.e., cost, maximum delay, and throughput. Our problem jointly considers the three metrics, while existing ones only account for one or two of them. We first show that our problem is uniquely challenging, as (i) it is NP-complete even to find a feasible solution satisfying all constraints, and (ii) directly extending existing solutions to our problem results in problem-dependent maximum delay violations that can be unbounded. We then design both an approximation algorithm and an efficient heuristic. For any feasible instance, our approximation algorithm will achieve a cost no worse than the optimal, while violating the maximum delay constraint and the throughput requirement only by constant ratios. Meanwhile, our heuristic will construct feasible solutions for a large portion (over 60% empirically) of feasible instances, strictly satisfying the maximum delay constraint and the throughput requirement. We further characterize a condition under which the cost of our heuristic must be within a problem-dependent-ratio gap to the optimal. We simulate representative edge computing platforms, and observe that (i) when sacrificing 3% throughput, our approximation algorithm reduces 32% cost as compared to a greedy baseline, and satisfies the maximum delay constraint for 56% simulated instances; (ii) our heuristic solves 62% of feasible instances, and reduces 24% cost as compared to the baseline while strictly satisfying all constraints.**

I. Introduction

We consider a multi-path network communication scenario where a sender streams a flow at a fixed rate to a receiver across a multi-hop network. Transmission over a link incurs a cost (e.g., energy consumption) and a delay (or latency equivalently), both of which are modeled as *arbitrary non-negative, non-decreasing, differentiable, and convex* functions of the link aggregate transmission rate. We study the multi-path routing problem that minimizes the network transmission cost subject to a maximum delay constraint and a throughput requirement. The maximum delay denotes the maximum Sender-to-Receiver (S2R) delay, or equivalently the delay of the slowest S2R path carrying traffic.

As a natural extension of the single-path routing for streaming a large volume of traffic while avoiding link traffic congestion, multi-path routing is known to be a basic paradigm of networking that can provide strictly better Quality of Service (QoS) than single-path routing [1], [2]. Our study is motivated by recent skyrocketing interests on leveraging the edge-cloud computing platform to support real-time computationally intensive IoT applications [3], [4], e.g., real-time object recognition on cellphones [5]. As in the discussions below, these IoT applications are sensitive to the three fundamental networking performance metrics including throughput, delay, and cost. The proximate computing on edge devices benefits QoS of IoT, including saving bandwidth, lowering response time, and reducing cost.

First, many IoT applications are bandwidth-hungry, posing certain *throughput* requirements. For example, in the intelligent driving scenario, a large volume of data is collected from various in-vehicle and on-board sensors to evaluate the driver's real-time driving performance, and must be processed in a timely manner to maintain the efficient and safe driving. As estimated in [21], by 2020, each autonomous vehicle may generate data at the rate of $20-40$ MB/second.

Second, real-time IoT applications are *delay*-sensitive. Consider the time-critical IoT control system in [22]. It leverages information from interaction between different operators, to perform remote control of robotic operations and collaborative robots in closed-loop control systems. The communication latency in this case shall be no more than 1 ms [22]. Another example is from [5]. Ran *et al.* [5] develop an Android IoT application of real-time object detection. In order to meet real-time latency requirements, the image processing rate is 9 FPS if we offload the application to a nearby server for processing, where the communication latency accounts for over 90% of the total latency.

In addition to throughput and delay, it is also critical to consider the *cost* for providing high-quality IoT services. For example, the massive IoT workloads in the edge/cloud lead to an enormous amount of energy consumption. The world's Information-Communication-Technologies ecosystem uses about 1500 TWh of electricity annually, equal to the combined electric generation of Japan and Germany [23]. It is very important to manage the cost of an edge-cloud computing system efficiently [24], as the electricity bill cost [25] is a significant part of its expenses. Here we remark that we can relate the processing cost (i.e., cost defined on a node) to the networking cost (i.e., cost defined on a link), by first replicating the node and then adding a link (with a traffic-dependent cost function) between the two replicated nodes.

Existing results. We study a multi-path routing problem,

TABLE I
COMPARISON OF OUR WORK AND RELATED STUDIES IN THE LITERATURE.

		Many, e.g., [6], [7]	[8]–[10]	[11]–[20]	Our Problem
Metrics	Aggregate S2R Cost	✗	✓	✗	✓
	Maximum S2R Delay	✗	✗	✓	✓
	Throughput	✓	✓	✓	✓
Hardness		Convex Program	Convex Program	NP-hard	NP-hard

with an objective of minimizing network transmission cost subject to both a maximum delay constraint and a throughput requirement. We compare our study with existing ones in Tab. I. First, many studies (e.g., [6], [7]) optimize throughput; and there are also studies (e.g., [8]–[10]) minimizing cost under throughput requirements. Their problems can be formulated as convex programs without considering the maximum delay constraint, and can be solved optimally in polynomial time. Next, several studies consider both maximum delay and throughput. Specifically, [11]–[18] minimize maximum delay under throughput requirements, and [19], [20] maximize throughput under maximum delay constraints (note that [15]–[20] consider a traffic-independent link delay model where each link has a constant delay, which is a special case of the traffic-dependent link delay model considered in our study and [11]–[14]). As maximum delay is non-convex, problems in those maximum-delay-aware studies are all NP-hard, hence cannot be solved optimally in polynomial time unless P = NP.

This paper takes the first step towards solving the multi-path routing problem that simultaneously considers three critical metrics of cost, maximum delay, and throughput. Solving our problem is uniquely challenging, as it is impossible even to find a feasible solution satisfying all constraints in polynomial time, unless P = NP (see Thm. 1). In comparison, existing studies only account for one or two of the three metrics (see Tab. I). Although existing maximum-delay-aware problems in [11]–[20] are all NP-hard, just obtaining feasible solutions are straightforward and can be done in polynomial time. Hence those existing studies can design efficient algorithms to achieve near-optimal feasible solutions meeting constraints.

There are some studies, e.g., [24], [26], [27], which jointly consider cost, delay, and throughput. However, they look at the average S2R delay instead of the maximum S2R delay. Note that maximum delay fundamentally differs from average delay: (i) maximum delay is non-convex, while average delay is convex [14], thus the maximum delay optimization is much more challenging than the average delay optimization; (ii) by the definition of maximum delay, it is clear that the solution with known maximum delay performance implies bounded S2R delay for all traffic, while the solution even with the minimal average delay performance suffers from a fatal limitation where certain traffic can experience an arbitrarily large S2R delay [14]. Therefore, in order to provide a low-delay routing service for all traffic, it is necessary to examine the maximum delay instead of the average delay.

Contributions. In this paper we design efficient algorithms to solve our multi-path routing problem. We summarize the theoretical performance guarantees of our algorithms in Tab. II, and make the following specific contributions:

▷ We prove that it is NP-complete even to find a feasible solution strictly satisfying all constraints, as our problem requires to jointly consider the three critical performance metrics, i.e., cost, maximum delay, and throughput. Thus it is fundamentally challenging to efficiently construct an approximate solution in polynomial time with bounded violations of constraints.

▷ For four well-known algorithms that only account for two of the three metrics, we prove that for any feasible instance, their achieved costs must be within problem-dependent-ratio gaps as compared to the optimal, after relaxing the maximum delay constraint also by problem-dependent ratios. However, those ratios can be unbounded in certain instances.

▷ By jointly considering the three metrics, we design a polynomial-time approximation algorithm. Theoretically, it is the first to achieve a cost no worse than the optimal after relaxing the maximum delay constraint and the throughput requirement by constant ratios, for all feasible instances. Empirically when sacrificing 3% throughput, it reduces 32% cost as compared to a greedy baseline and satisfies the maximum delay constraint for 56% simulated instances.

▷ We develop an efficient heuristic, which can solve a large portion of feasible instances, strictly satisfying the maximum delay constraint and the throughput requirement. We further characterize a condition under which the cost of our heuristic must be within a problem-dependent-ratio gap to the optimal. Empirically, our heuristic solves 62% of feasible instances, and reduces 24% cost than the baseline, meeting all constraints.

II. PROBLEM DEFINITION

We consider a multi-hop network modeled as a directed graph $G \triangleq (V, E)$ where V is the set of nodes and E is the set of links. Data transmission over a link $e \in E$ incurs both a delay modeled as a function $d_e(x_e)$ and a cost modeled as a function $c_e(x_e)$, where x_e is the aggregate link traffic rate. We consider a fundamental network communication scenario where a sender $s \in V$ streams a flow at a rate of $R > 0$ to a receiver $t \in V \backslash \{s\}$, possibly using multiple paths.

Due to practical concerns, we assume that both $d_e(x_e)$ and $c_e(x_e)$ are *arbitrary functions of* $x_e \geq 0$ *that are non-negative, non-decreasing, differentiable, and convex.* We also assume that $c_e(x_e)$ is positive if $x_e > 0$. Representative examples are presented in, but not restricted to, the following.

Delay function example. In many cases where the traffic to be streamed is large but the networking resources are limited, the queuing delay dominates the networking delay. Assuming

TABLE II
SUMMARY OF THEORETICAL PERFORMANCE GUARANTEES OF OUR PROPOSED ALGORITHMS.

Proposed Solution		f_{DSO}	f_{CSO}	f_{DNE}	f_{CNE}	Algorithm 1	Algorithm 2*
Proof to its Performance Bound		Thm. 2	Thm. 2	Thm. 3	Thm. 3	Thm. 4	Thm. 5 and Thm. 6
Approximation Ratio	Aggregate S2R Cost	$\mu\cdot\nu$	1	$\mu\cdot\nu\cdot\alpha_D$	α_C	1	$\mu\cdot\nu^{**}$
Relaxation Ratio	Maximum S2R Delay	γ_D	$\mu\cdot\nu\cdot\gamma_C$	α_D	$\mu\cdot\nu\cdot\alpha_C$	$1/\epsilon$	1^*
	Throughput	1	1	1	1	$1-\epsilon$	1^*

Note: μ and ν are defined in Lem. 6. γ_D and γ_C are defined in Lem. 4, Appendix. α_D and α_C are defined in Lem. 5, Appendix. We remark that μ, ν, α_D, α_C, γ_D, and γ_C are all problem-dependent, but ϵ is independent to problem instances and can be any constant in the range of $(0,1)$.
*: Algorithm 2 solves a large portion of feasible instances, while all the other algorithms can solve every feasible instance.
**: The approximation ratio of Algorithm 2 holds only under our derived condition presented in Thm. 6.

M/M/1 queue together with a FIFO server, according to queuing theory, the link delay can be estimated as follows [28]

$$d(x) \;=\; \begin{cases} \frac{1}{v-x}, & \text{if } v > x; \\ +\infty, & \text{otherwise}, \end{cases} \tag{1}$$

where v is the link capacity and x is the assigned traffic. More complex delay functions have been proposed to account for the delay when $x \geq v$ [29]. Overall, it is reasonable to assume non-negative, non-decreasing, differentiable, and convex delay functions.

Cost function example. Here we take the power consumption as an example. Yu *et al.* [27] propose to use a linear model to calculate the power consumption $c(x)$ of an edge device with allocated workloads x as follows

$$c(x) = \left(q^{\mathsf{idle}} + \left(q^{\mathsf{peak}} - q^{\mathsf{idle}}\right)\cdot x/v\right), \tag{2}$$

where q^{idle} (resp. q^{peak}) is the idle power (resp. peak power). Alternatives are also proposed, e.g., a quadratic function [24]. Overall, they meet our assumptions on cost functions.

We denote $\mathcal{P}$ as the set of paths from s to t. A solution f of our problem is a network flow from s to t and defined as the assigned flow rates over $\mathcal{P}$, i.e., $f \triangleq \{x^p : x^p \geq 0, \forall p \in \mathcal{P}\}$, where x^p is the assigned flow rate on the path p. We define x_e as the aggregate flow rate of the link $e \in E$, and we have

$$x_e \;\triangleq\; \sum_{p\in\mathcal{P}:e\in p} x^p.$$

The delay of the path p under flow f is defined as

$$d^p(f) \;\triangleq\; \sum_{e\in E:e\in p} d_e(x_e),$$

which is the sum of link delay for all links belonging to p. Similarly, we can define the cost of the path p, i.e., $c^p(f)$, as the sum of link cost for all links belonging to p.

The total delay of f is defined by

$$\mathcal{T}(f) \;\triangleq\; \sum_{p\in\mathcal{P}} d^p(f)\cdot x^p \;=\; \sum_{e\in E} d_e(x_e)\cdot x_e. \tag{3}$$

The total cost $\mathcal{C}(f)$ of f is defined in a way similar to (3) but with respect to the link cost.

Different from the total delay, the maximum delay of f is

$$\mathcal{M}(f) \;\triangleq\; \max_{p\in\mathcal{P}:x^p>0} d^p(f), \tag{4}$$

which is the delay of the slowest path that is assigned a positive rate. We can define the maximum cost $\mathcal{N}(f)$ in a way similar to (4) but with respect to the link cost.

Given a feasible problem instance, we define $\mathcal{T}^*$ (resp. $\mathcal{M}^*$) as the minimal total delay (resp. minimal maximum delay) that can be achieved by any network flow solution supporting a rate of R, regardless of the achieved cost. Similarly, we define $\mathcal{C}^*$ (resp. $\mathcal{N}^*$) as the minimal total cost (resp. minimal maximum cost) under a rate of R, regardless of the experienced delay.

We define the total flow rate sent by f from s to t, or equivalently the throughput of f, as

$$|f| \;\triangleq\; \sum_{p\in\mathcal{P}} x^p \;=\; \sum_{e\in\mathsf{Out}(s)} x_e \;=\; \sum_{e\in\mathsf{In}(t)} x_e,$$

where $\mathsf{Out}(v) \triangleq \{(v,u) \in E : \forall u \in V\backslash\{v\}\}$ is the set of outgoing links of $v \in V$, and similarly $\mathsf{In}(v) \triangleq \{(u,v) \in E : \forall u \in V\backslash\{v\}\}$ is the set of incoming links of $v \in V$. We define the average delay $\mathcal{A}(f)$ of f as $\mathcal{A}(f) = \mathcal{T}(f)/|f|$.

We study the maximum-Delay-constrained Throughput-guaranteed Cost minimization problem (DTC) below:

$$(\mathsf{DTC}): \;\text{obj.}\; \min_{f\in F}\; \mathcal{C}(f) \tag{5a}$$

$$\text{s.t.}\;\; \mathcal{M}(f) \;\leq\; D, \;\; \forall\, f \in F, \tag{5b}$$

$$|f| \;=\; R, \;\; \forall\, f \in F, \tag{5c}$$

where F is the set of all feasible network flows from s to t. The objective in (5a) minimizes the total cost. The constraint in (5b) is the maximum delay constraint restricting that the $\mathsf{S2R}$ delay of all traffic no larger than a constant $D \geq 0$. The constraint in (5c) is the throughput requirement.

We note that Correa *et al.* [12] study a maximum delay minimization problem, i.e., $\min \mathcal{M}(f)$ subject to $|f| = R$. Their problem only considers maximum delay and throughput but not cost. It is proven to be NP-hard [12]. Now by adapting its NP-hardness proof, in the following we give a theorem to present the computational complexity of DTC.

Theorem 1: For DTC, deciding whether there is a feasible solution strictly satisfying all the constraints is NP-complete.

Proof: It is a straightforward adaptation of the proof of [12, Thm. 3.3], and is skipped due to page limit. ■

III. EXTENDING EXISTING SOLUTIONS TO OUR PROBLEM

For existing maximum-delay-aware studies (see Tab. I), we note that (i) studies [15]–[20] assume that each link has a constant delay, which is a special case of our traffic-dependent link delay model. It is unclear how to generalize their technique of time-expanded networks to our problem; (ii) studies [11]–[14] model link delay as traffic-dependent functions, the same as our model. They focus on studying two kinds of well-known network flow solutions, i.e., the system-optimal flow and the Nash-equilibrium flow. In this section, we prove that the delay-/cost- system-optimal flow and the delay-/cost- Nash-equilibrium flow are all approximate solutions to DTC, providing problem-dependent approximation ratios after relaxing the maximum delay constraint also by problem-dependent ratios. We remark that these ratios can be unbounded in certain instances.

First we briefly define these well-know network flow solutions (detailed definitions are referred to [12]).

Delay-/cost- system-optimal flow. The delay-system-optimal flow is a single-unicast flow, minimizing total delay subject to a throughput requirement. Similarly, the cost-system-optimal flow minimizes total cost subject to a throughput requirement.

With our models on delay functions and cost functions, both system-optimal flows can be achieved in polynomial time [12]. We denote the delay-system-optimal flow by f_{DSO}, and denote the cost-system-optimal flow by f_{CSO}. According to the study [12], theoretically there is a performance gap of γ_D (resp. γ_C) comparing $\mathcal{M}(f_{\mathsf{DSO}})$ (resp. $\mathcal{N}(f_{\mathsf{CSO}})$) to $\mathcal{M}^*$ (resp. $\mathcal{N}^*$). Both γ_D and γ_C depend on problem instances. We introduce them in Lem. 4, Appendix.

Delay-/cost- Nash-equilibrium flow. A single-unicast flow f subject to a throughput requirement R is a delay-Nash-equilibrium flow, if and only if $|f| = R$, and for any pair of paths $p_1 \in \mathcal{P}$ and $p_2 \in \mathcal{P}$ with $x^{p_1} > 0$, it must hold that $d^{p_1}(f) \le d^{p_2}(f)$. Similarly, we can define a cost-Nash-equilibrium flow in terms of $c^p(f)$ instead of $d^p(f)$.

According to [12], the delay-Nash-equilibrium flow can be achieved in polynomial time with our delay function assumptions, and we denote it by f_{DNE}. Similar result holds for the cost-Nash-equilibrium flow, which is denoted by f_{CNE}. Similar to the system-optimal flows, theoretically there is a problem-dependent performance gap of α_D comparing $\mathcal{M}(f_{\mathsf{DNE}})$ (resp. $\mathcal{T}(f_{\mathsf{DNE}})$) to $\mathcal{M}^*$ (resp. $\mathcal{T}^*$), and a problem-dependent performance gap of α_C comparing $\mathcal{N}(f_{\mathsf{CNE}})$ (resp. $\mathcal{C}(f_{\mathsf{CNE}})$) to $\mathcal{N}^*$ (resp. $\mathcal{C}^*$) [12]. We introduce α_D and α_C in Lem. 5, Appendix.

Now we define the following two constants μ and ν, by comparing link delay functions with link cost functions.

Lemma 1: Comparing the link cost function $c_e(x)$ with the link delay function $d_e(x)$, we define μ and ν to be the minimum number that meets the following constraints:

$$c_e(x) \le \mu \cdot d_e(x), d_e(x) \le \nu \cdot c_e(x) : \forall e \in E, \forall x \in (0, R].$$

The following theorems characterize the performance gaps comparing the system-optimal flows or the Nash-equilibrium flows to the optimal solution of DTC.

Theorem 2: Suppose f_{OPT} is the optimal solution of DTC. Then f_{DSO} must exist, and it holds that

$$\mathcal{C}(f_{\mathsf{DSO}}) \le \mu \cdot \nu \cdot \mathcal{C}(f_{\mathsf{OPT}}),\ |f_{\mathsf{DSO}}| = R,\ \mathcal{M}(f_{\mathsf{DSO}}) \le \gamma_D \cdot D.$$

f_{CSO} must exist, too, and it holds that

$$\mathcal{C}(f_{\mathsf{CSO}}) \le \mathcal{C}(f_{\mathsf{OPT}}),\ |f_{\mathsf{CSO}}| = R,\ \mathcal{M}(f_{\mathsf{CSO}}) \le \gamma_C \cdot \mu \cdot \nu \cdot D.$$

Proof: Refer to Appendix. ∎

Theorem 3: Suppose f_{OPT} is the optimal solution of DTC. Then f_{DNE} must exist, and it holds that

$$\mathcal{C}(f_{\mathsf{DNE}}) \le \mu\cdot\nu\cdot\alpha_D\cdot\mathcal{C}(f_{\mathsf{OPT}}),\ |f_{\mathsf{DNE}}| = R,\ \mathcal{M}(f_{\mathsf{DNE}}) \le \alpha_D\cdot D.$$

f_{CNE} must exist, too, and it holds that

$$\mathcal{C}(f_{\mathsf{CNE}}) \le \alpha_C\cdot\mathcal{C}(f_{\mathsf{OPT}}),\ |f_{\mathsf{CNE}}| = R,\ \mathcal{M}(f_{\mathsf{CNE}}) \le \mu\cdot\nu\cdot\alpha_C\cdot D.$$

Proof: Refer to Appendix. ∎

The above theorems suggest that the costs of the system-optimal flows and the Nash-equilibrium flows are within problem-dependent-ratio gaps to the optimal, after relaxing the maximum delay constraint by problem-dependent ratios. As discussed in [12], [14], there exist certain feasible instances where the system-optimal flows and the Nash-equilibrium flows violate the maximum delay constraint by arbitrarily large ratios, i.e., γ_D, γ_C, α_D, α_C can be unbounded.

IV. An Approximation Algorithm

In this section we develop a polynomial-time approximation algorithm for DTC. By jointly considering cost, maximum delay, and throughput, for any feasible instance of DTC, it achieves a cost no worse than the optimal, after relaxing both the maximum delay constraint and the throughput requirement by constant ratios independent to problem instances.

For DTC, the maximum delay constraint is non-convex and hence makes the problem challenging to be solved. Now we replace the non-convex maximum delay by the convex average delay, and solve the Average-delay-constrained Throughput-guaranteed Cost minimization problem (ATC) instead:

$$(\mathsf{ATC}) : \text{obj.} \min_{f \in F} \mathcal{C}(f) \tag{6a}$$
$$\text{s.t. } \mathcal{T}(f) \ \le\ D \cdot R, \ \ \forall\ f \in F, \tag{6b}$$
$$|f| \ =\ R, \ \ \forall\ f \in F. \tag{6c}$$

We use an algorithm with a similar structure to [14, Algorithm 1] to solve DTC approximately in polynomial time (see Algorithm 1). We first obtain a path-based optimal flow solution of ATC by solving problem (6) with an edge-based flow formulation and then do flow decomposition. Next

Algorithm 1 Proposed Approximation Algorithm

```
procedure
    f = ATC(G, R, D, s, t)
    x^delete = ε · R
    while x^delete > 0 do
        Find the slowest flow-carrying path p_l of f
        if x^{p_l} > x^delete then
            x^{p_l} = x^{p_l} − x^delete, x^delete = 0
        else
            x^delete = x^delete − x^{p_l}, x^{p_l} = 0
    return the remaining flow f
```

we delete flow rate of $\epsilon \cdot R$ iteratively from its slowest flow-carrying paths. The remaining flow in the end is an approximate solution to DTC, which is proven in our Thm. 4. Algorithm 1 has a polynomial time complexity, because (i) problem (6) can be formulated as a convex program that has a polynomial size, using the edge-based flow formulation (i.e., using variables x_e), (ii) it takes a polynomial time to do flow decomposition [30], (iii) flow decomposition outputs at most $|E|$ flow-carrying paths [30], hence leading to $O(|E|)$ number of iterations to delete flow rate, and (iv) it takes $O(|E|)$ time to update delays of the remaining utilized paths after deleting rate from the current slowest flow-carrying path in each iteration.

Lemma 2: In Algorithm 1, suppose f_A is the optimal solution of ATC (the flow f in Line 2), and f_B is the solution returned in the end (the flow f in Line 10). We must have

$$\mathcal{T}(f_B) + \epsilon \cdot R \cdot \mathcal{M}(f_B) \quad \leq \quad \mathcal{T}(f_A).$$

Proof: It is similar to the proof of [14, Lem. 1], and is skipped due to page limit. ■

With Lem. 2, we can prove the performance guarantee of out Algorithm 1 in the following theorem.

Theorem 4: Suppose f_{OPT} is the optimal solution of DTC. Then Algorithm 1 with an arbitrary $\epsilon \in (0, 1)$ must return a flow f, with the following held

$$\mathcal{C}(f) \ \leq \ \mathcal{C}(f_{\mathsf{OPT}}), \ \ |f| \ \geq \ (1-\epsilon) \cdot R, \ \ \mathcal{M}(f) \ \leq \ D/\epsilon.$$

Proof: It is clear that $|f_{\mathsf{OPT}}| = R$ and $\mathcal{M}(f_{\mathsf{OPT}}) \leq D$. Now let us consider the following inequality

$$\mathcal{T}(f_{\mathsf{OPT}}) \ \leq \ \mathcal{M}(f_{\mathsf{OPT}}) \cdot |f_{\mathsf{OPT}}| \ \leq \ D \cdot R.$$

Then we know that f_{OPT} is also a feasible solution of ATC. Hence our Algorithm 1 must return a flow solution.

Because f_A is optimal to ATC, we have $\mathcal{C}(f_A) \leq \mathcal{C}(f_{\mathsf{OPT}})$. Due to the non-decreasing property of link cost function, considering $f = f_B$ is the flow after we delete rate from f_A, we have $\mathcal{C}(f) \leq \mathcal{C}(f_A)$, implying that $\mathcal{C}(f) \leq \mathcal{C}(f_{\mathsf{OPT}})$.

As for the maximum delay performance of f, we have

$$\mathcal{M}(f) \overset{(a)}{\leq} \frac{\mathcal{T}(f_A) - \mathcal{T}(f_B)}{\epsilon \cdot R} \leq \frac{\mathcal{T}(f_A)}{\epsilon \cdot R} \overset{(b)}{\leq} \frac{R \cdot D}{\epsilon \cdot R} = \frac{D}{\epsilon},$$

where inequality (a) comes from Lem. 2, inequality (b) holds because f_A is a feasible solution of ATC.

And it holds that $|f| \geq (1-\epsilon) \cdot R$, because that we delete $\epsilon \cdot R$ rate from f_A where $|f_A| = R$ to obtain f. ■

Based on Thm. 4, for every feasible instance of DTC, Algorithm 1 must achieve a cost no worse than the optimal, after relaxing the maximum delay constraint by a ratio of $(1/\epsilon)$ and relaxing the throughput requirement by a ratio of $(1-\epsilon)$. Here $\epsilon \in (0, 1)$ is an arbitrary user-defined number and independent to instances. We emphasize again that although Algorithm 1 obtains the optimal cost by violating constraints by bounded ratios, it is valuable, as it is NP-complete even to find a feasible solution meeting all constraints for DTC.

Algorithm 2 Proposed Heuristic Approach

```
procedure
    Enumerate r ≥ 0 to figure out a r* such that |f_{r*}| ≥ R
    f = f_{r*}, x^delete = |f_{r*}| − R
    while x^delete > 0 do
        Find the path p_c with the largest path cost among all
        flow-carrying paths of f
        if x^{p_c} > x^delete then
            x^{p_c} = x^{p_c} − x^delete, x^delete = 0
        else
            x^delete = x^delete − x^{p_c}, x^{p_c} = 0
    return the remaining flow f
```

V. An Efficient Heuristic Approach

We introduce multiple algorithms to solve DTC approximately in Sec. III and Sec. IV. However, theoretically they all can violate the maximum delay constraint or the throughput requirement. In this section, we design an efficient heuristic approach for DTC. We prove that the solution of our heuristic must be feasible, strictly satisfying all constraints.

We note that the non-convex maximum delay constraint is the main challenge of solving DTC efficiently. In Sec. IV, our Algorithm 1 deals with the challenge by replacing the maximum delay by the average delay that is convex, and then solving the average-delay-constrained counterpart of DTC. Here we propose another way to handle the challenge, i.e., solving the following problem which can be casted as a convex program instead of directly solving the DTC:

$$\text{obj.} \quad \min_{f \in F} \ \mathcal{C}(f) \tag{7a}$$

$$\text{s.t.} \quad |f| - \mathcal{T}(f)/D \ \geq \ r, \ \forall \, f \in F, \tag{7b}$$

where $r \geq 0$ is a user-defined constant. We denote f_r as the optimal solution of above problem given an input r. We introduce a critical observation of f_r in the following lemma.

Lemma 3: Given any $r \geq 0$, f_r can be obtained in polynomial time, with the following held

$$\mathcal{M}(f_r) \ \leq \ D.$$

Proof: Refer to Appendix. ■

Lem. 3 suggests that the optimal solution of the problem formulated in (7) can be figured out quickly, and must satisfy the maximum delay constraint of DTC. Therefore, in order to obtain feasible solutions of DTC meeting both the maximum delay constraint and the throughput requirement, it is a natural idea of enumerating r and solving the associated problem formulated in (7) iteratively till we figure out a solution that satisfies the throughput requirement. Following this idea, we design a heuristic approach for DTC in Algorithm 2, and prove that our heuristic solution must be feasible to DTC in Thm. 5.

Theorem 5: If Algorithm 2 obtains a network flow solution f, the following must hold

$$|f| \ = \ R, \ \ \mathcal{M}(f) \ \leq \ D.$$

Proof: It is a direct result of our Lem. 3, together with the details of Algorithm 2. ■

Next we characterize a sufficient condition under which our heuristic solution must be approximate to DTC, with theoretical performance guarantee.

Theorem 6: Suppose f_{OPT} is the optimal solution of DTC. If Algorithm 2 obtains a network flow solution f with $f = f_{r^*}$, i.e., with $|f_{r^*}| = R$ in Line 2 of Algorithm 2, then the following must hold

$$\mathcal{C}(f) \ \leq \ \mu \cdot \nu \cdot \mathcal{C}(f_{\mathsf{OPT}}).$$

Proof: Refer to Appendix. ■

Based on above theorems, our heuristic solution must be feasible to DTC; moreover, it must be approximate to DTC under our derived condition. However, our heuristic may not figure out a flow solution for certain feasible instances of DTC, because Algorithm 2 may not find a f_{r^*} with $|f_{r^*}| \geq R$ in Line 2. As evaluated later in Sec. VI, empirically our heuristic solves a large portion of all feasible instances. In comparison, theoretically all of our proposed algorithms in Sec. III and Sec. IV must obtain solutions for all feasible instances of DTC, in spite of violating constraints by certain ratios.

Finally, we remark that for instances where Algorithm 2 cannot find a f_{r^*} with $|f_{r^*}| \geq R$ in Line 2, we can use the f_r enumerated by Algorithm 2 achieving the largest throughput as a solution to DTC, meeting the maximum delay constraint but violating the throughput requirement.

VI. Performance Evaluation

We evaluate our solutions by simulating edge computing platforms where multiple edge computing nodes (i.e., devices that can process IoT applications) are connected with access point nodes (i.e., devices that can directly receive data from IoT applications). We consider two platforms with representative network topologies (see Fig. 1). (i) One is a complete binary tree with 15 nodes. It logically represents a hierarchical edge computing structure. We assume the leaves are access points, while all interior nodes are edge computing devices. (ii) The other is a square grid with 36 nodes, 30 horizontal links, and 30 vertical links. This topology represents a distributed edge computing structure, and has been used in existing edge computing studies, e.g., in [31]. We assume the four corner nodes are access points, while all the remaining nodes are edge computing devices. Both platforms are modeled as undirected graphs, where each undirected link is treated as two directed links that operate independently, a common way to model an undirected graph by a directed one, e.g. in [14]. In each platform, we assume a virtual sender s is connected with every access point node, and a virtual receiver t is connected with every edge computing node. Then a flow from s to t represents a routing solution following which we can offload and distribute IoT workloads from IoT devices to edge servers, across access points and the associated platform.

We consider Equation (1) as the link delay function and Equation (2) as the link cost function. For each $e \in E$, we randomly select its capacity from $\{10, 20, 30, 40, 50\}$Mbps,

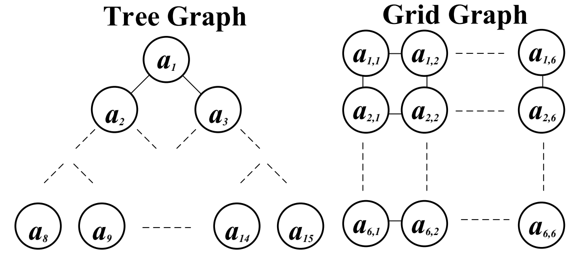


Fig. 1. Simulated edge computing platforms.

similar to [31]. Following [27], the peak power cost q_e^{peak} is randomly chosen from $\{100, 200, 300, 400, 500\}$Watts, and the idle power cost is set as $q_e^{\mathsf{idle}} = q_e^{\mathsf{peak}}/2$. We set D as $D = 200$ms. Our test environment is an Intel Core i5 2.40GHz processor with 8GB memory. Convex programs are solved by CVX [32], and all other experiments are implemented in C++.

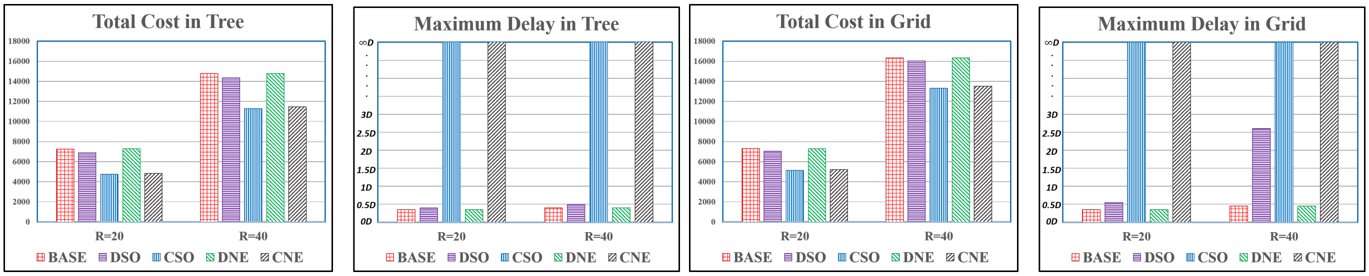
We compare our solutions with a *baseline*. We hope that the baseline can quickly obtain feasible solutions for every feasible instance of DTC in simulations, satisfying the maximum delay constraint and the throughput requirement. This is already theoretically challenging based on Thm. 1. We consider a conceivable greedy approach to obtain throughput-guaranteed maximum-delay-minimal flow solutions. It allocates a small fraction of the throughput R, i.e., $\delta \cdot R$, to the fastest S2R path iteratively, til all R is satisfied. We set $\delta = 1\%$. The baseline provides a rough approximation to the minimal maximum delay M^* subject to a throughput requirement R. We compare M^* against D to find out if the simulated instance is feasible.

Simulation results of f_{DSO}, f_{CSO}, f_{DNE}, and f_{CNE}. We first compare the baseline with solutions f_{DSO}, f_{CSO}, f_{DNE}, and f_{CNE}. We consider four scenarios where the throughput requirement R is either 20 or 40, and the edge computing platform is either tree or grid. Fig. 2 shows the cost and maximum delay averaged over 1000 simulations.

According to Fig. 2, the first observation is that the baseline performs almost the same as f_{DNE}. Second, it is expected that the costs of the baseline, f_{DSO}, and f_{DNE} are much worse (30% worse in the tree, and 20% worse in the grid) than those of f_{CSO} and f_{CNE}, since no cost optimization is involved in the baseline, f_{DSO}, and f_{DNE}. Similarly, since no delay optimization is involved in f_{CSO} and f_{CNE}, among 1000 simulations of each experimental scenario, there exist problem instances where f_{CSO} and f_{CNE} will assign a flow rate x_e that exceeds v_e for certain link $e \in E$, leading to an infinitely large delay according to the queuing delay function (1).

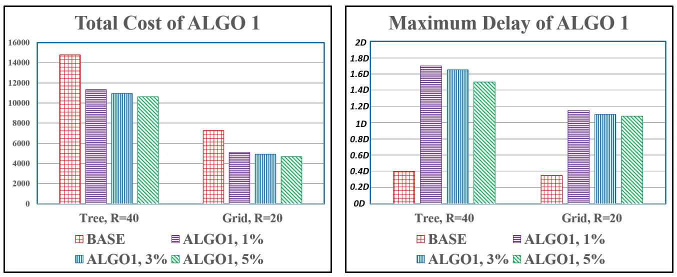
Simulation results of Algorithm 1. We now evaluate Algorithm 1 by respectively running 1000 simulations on two experimental scenarios, one using the tree given $R = 40$ while the other using the grid given $R = 20$. Fig. 3 presents simulation results of Algorithm 1 with a very small ϵ (1%, 3%, and 5% respectively) as well as the baseline approach.

According to Fig. 3(a), on average Algorithm 1 reduces cost substantially (over 20% for the tree, and over 30% for the grid) than the baseline; and according to Fig. 3(b), Algorithm 1 violates the maximum delay constraint by a ratio that is way



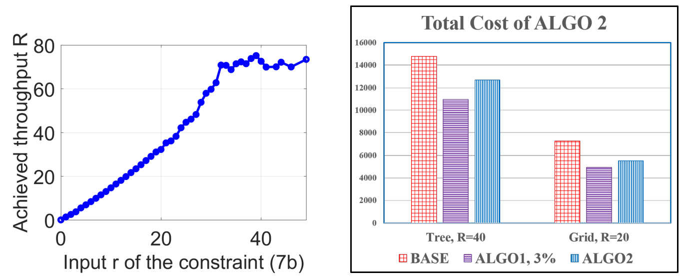
(a) Total cost in the tree. (b) Maximum delay in the tree. (c) Total cost in the grid. (d) Maximum delay in the grid.

Fig. 2. Averaged simulation results of f_{DSO}, f_{CSO}, f_{DNE}, and f_{CNE}. The maximum delays of f_{CSO} and f_{CNE} are infinitely large as compared to D, since they can assign a flow rate which exceeds v_e for certain link $e \in E$, but the queuing delay function (1) does not allow assigned traffic to be larger than v_e.



(a) Total cost results. (b) Maximum delay results.

Fig. 3. Averaged simulation results of the baseline, Algorithm 1 with $\epsilon = 1\%$, with $\epsilon = 3\%$, and with $\epsilon = 5\%$, respectively.



(a) The throughput achieved by the optimal solution of problem (7), with r that is an input of problem (7).

(b) Comparing the cost of Algorithm 2 respectively with that of the baseline and of our Algorithm 1.

Fig. 4. Simulation results of our heuristic Algorithm 2.

TABLE III
RATIO OF SIMULATED INSTANCES WHERE ALGORITHM 1 STRICTLY SATISFIES THE MAXIMUM DELAY CONSTRAINT.

	Baseline	Algo1, 1%	Algo1, 3%	Algo1, 5%
$R = 40$, Tree	100%	18%	22%	32%
$R = 20$, Grid	100%	50%	56%	58%

TABLE IV
ESTIMATION OF THE PORTION OF FEASIBLE INSTANCES THAT CAN BE SOLVED BY OUR HEURISTIC ALGORITHM 2.

	Mean	Min	1st-Quartile	Median	3rd-Quartile	Max
Tree	72%	67%	71%	72%	73%	79%
Grid	62%	50%	58%	62%	65%	72%

smaller than its theoretical counterpart $(1/\epsilon)$. In addition, we observe that Algorithm 1 in fact meets the maximum delay constraint for a large portion of simulated instances (see Tab. III). Moreover, Algorithm 1 never assign a flow rate which exceeds v_e to a link $e \in E$ in all simulated instances, which is different from solutions f_{CSO} and f_{CNE}.

Simulation results of Algorithm 2. Now we evaluate our heuristic Algorithm 2. First we look at the achieved throughput of the optimal solution of problem (7) (the problem solved by our heuristic) with respect to r, with the simulation results of one instance given in Fig. 4(a). We observe that the achieved throughput is increasing with r when r is small/medium, but is non-monotonic with r when r is large. We leave it as a future direction of characterizing conditions under which the achieved throughput is monotonic with the input r.

Next we evaluate the cost of Algorithm 2, as compared to that of the baseline and of Algorithm 1 ($\epsilon = 3\%$). For the enumeration (Line 2) of Algorithm 2, we increase r from 0 with a step of 1, and set r^* to be the first r which achieves a throughput no smaller than the requirement R. We simulate 1000 instances on the tree (resp. grid) assuming $R = 40$ (resp. $R = 20$), with the results illustrated in Fig. 4(b). We observe that (i) both the baseline and Algorithm 2 obtain feasible solutions for all simulated instances, meeting all constraints, and Algorithm 2 reduces cost by 15% (resp. 24%) than the baseline for instances of the tree (resp. instances of the grid); (ii) Algorithm 1 reduces cost by 26% (resp. 32%) than the baseline when sacrificing 3% throughput requirement, and satisfies the maximum delay constraint for 22% (resp. 56%) simulated instances of the tree (resp. of the grid).

Note that theoretically Algorithm 2 can only solve a portion of feasible instances of DTC. The reason why it solves all simulated instances in Fig. 4(b) is that $R = 40$ (resp. $R = 20$) is not a large throughput requirement for the tree platform (resp. grid platform) given $D = 200$. Hence, we further estimate the portion of feasible instances solved by Algorithm 2 with comprehensive simulations. Given a platform subject to $D = 200$, we use simulation to figure out (i) the maximum throughput achieved by feasible flows, denoted by R_b, and (ii) the maximum throughput achieved by solving the problem (7) after enumerating r, denoted by R_h. We use R_h/R_b to estimate the portion of feasible instances solved by Algorithm 2, and the results are given in Tab. IV after running 1000 simulations on the tree platform and on the grid platform respectively. As in the table, our heuristic can solve a large portion of feasible instances of DTC.

VII. Conclusion and Future Work

We consider the scenario where a sender streams a flow at a fixed rate to a receiver across a multi-hop network, possibly using multiple paths. Data transmission over a link incurs a delay and a cost both of which are traffic-dependent. We study the problem of minimizing network transmission cost under constraints on the maximum delay and the throughput. Our study is important for leveraging edge-cloud computing platforms to support computationally intensive IoT applications. The need to jointly consider cost, maximum delay, and throughput differentiates our problem from existing ones. We prove it is NP-complete even to find a feasible solution, and direct extensions of existing algorithms obtain solutions that theoretically violate the maximum delay constraint by problem-dependent ratios (can be unbounded in certain instances). It is thus uniquely challenging to solve our problem.

We design both an approximation algorithm and an efficient heuristic. Our approximation algorithm can solve every feasible instance and will achieve a cost no worse than the optimal, after relaxing both the maximum delay constraint and the throughput requirement by constant ratios independent to instances. Our heuristic approach can solve a large portion of feasible instances, satisfying both the maximum delay constraint and the throughput requirement. We further characterize a condition under which the cost of our heuristic must be within a problem-dependent-ratio gap to the optimal.

We leave it as an important future direction of theoretically characterizing conditions on instances under which our heuristic is guaranteed to solve those instances.

Acknowledgment

This work is partially funded by NSF Grants No. 1812963 and 1837519.

Appendix

Lemma 4: [12] With γ_D defined as the minimal positive number satisfying the following constraint:

$$d_e(x) + x \cdot d_e'(x) \leq \gamma_D \cdot d_e(x), \forall e \in E, \forall x \in [0, R],$$

and γ_C defined similarly but with respect to $c_e(x)$ instead of $d_e(x)$, the following result holds theoretically:

$$\mathcal{M}(f_{\mathsf{DSO}}) \leq \gamma_D \cdot \mathcal{M}^*, \ \mathcal{N}(f_{\mathsf{CSO}}) \leq \gamma_C \cdot \mathcal{N}^*$$

Moreover, the following holds for f_{DSO}

$$d^{p_1}(f_{\mathsf{DSO}})/d^{p_2}(f_{\mathsf{DSO}}) \leq \gamma_D, \forall p_1, p_2 \in \mathcal{P} : x^{p_1} > 0, x^{p_2} > 0.$$

Similarly, the following holds for f_{CSO}

$$c^{p_1}(f_{\mathsf{CSO}})/c^{p_2}(f_{\mathsf{CSO}}) \leq \gamma_C, \forall p_1, p_2 \in \mathcal{P} : x^{p_1} > 0, x^{p_2} > 0.$$

Lemma 5: [12] With α_D defined bellow

$$\alpha_D = \left(1 - \sup_{e\in E, 0\leq x\leq R} \left\{ \frac{x \cdot (d_e(R) - d_e(x))}{R \cdot d_e(R)} \right\}\right)^{-1},$$

and α_C defined similarly but with respect to $c_e(x)$ instead of $d_e(x)$, the following results hold theoretically:

$$\mathcal{M}(f_{\mathsf{DNE}}) \leq \alpha_D \cdot \mathcal{M}^*, \ \mathcal{T}(f_{\mathsf{DNE}}) \leq \alpha_D \cdot \mathcal{T}^*.$$
$$\mathcal{N}(f_{\mathsf{CNE}}) \leq \alpha_C \cdot \mathcal{N}^*, \ \mathcal{C}(f_{\mathsf{CNE}}) \leq \alpha_C \cdot \mathcal{C}^*.$$

Lemma 6: With our definition of μ and ν, the following holds theoretically for any single-unicast flow f

$$\mathcal{T}(f) \leq \nu \cdot \mathcal{C}(f), \ \mathcal{C}(f) \leq \mu \cdot \mathcal{T}(f).$$

Proof: It is a direct result of the following inequalities:

$$\mathcal{T}(f) = \sum_{e\in E}(d_e(x_e)\cdot x_e) \leq \nu \cdot \sum_{e\in E}(c_e(x_e)\cdot x_e) = \nu\cdot\mathcal{C}(f).$$
$$\mathcal{C}(f) = \sum_{e\in E}(c_e(x_e)\cdot x_e) \leq \mu \cdot \sum_{e\in E}(d_e(x_e)\cdot x_e) = \mu\cdot\mathcal{T}(f).$$

■

Proof of Thm. 2.

Proof: Since f_{OPT} is a feasible solution to the problem of finding the system-optimal flow, the existence of f_{OPT} implies the existence of f_{DSO} and f_{CSO}, where $|f_{\mathsf{DSO}}| = |f_{\mathsf{CSO}}| = R$.

As for the maximum delay of f_{DSO}, we have

$$\mathcal{M}(f_{\mathsf{DSO}}) \overset{(a)}{\leq} \gamma_D \cdot \mathcal{M}^* \leq \gamma_D \cdot \mathcal{M}(f_{\mathsf{OPT}}) \leq \gamma_D \cdot D,$$

where inequality (a) comes from Lem. 4.

As for the total cost of f_{DSO}, we have

$$\mathcal{C}(f_{\mathsf{DSO}}) \overset{(a)}{\leq} \mu\cdot\mathcal{T}(f_{\mathsf{DSO}}) \overset{(b)}{\leq} \mu\cdot\mathcal{T}(f_{\mathsf{OPT}}) \overset{(c)}{\leq} \mu\cdot\nu\cdot\mathcal{C}(f_{\mathsf{OPT}}),$$

where inequalities (a) and (c) comes from Lem. 6, and inequality (b) holds due to that f_{DSO} minimizes the total delay.

By the definition of f_{CSO}, first it is straightforward that $\mathcal{C}(f_{\mathsf{CSO}}) \leq \mathcal{C}(f_{\mathsf{OPT}})$. As for its maximum delay, we have

$$\begin{aligned}\mathcal{M}(f_{\mathsf{CSO}}) &= d^{p'}(f_{\mathsf{CSO}}) \leq \nu\cdot c^{p'}(f_{\mathsf{CSO}}) \overset{(a)}{\leq} \gamma_C\cdot\nu\cdot c^{p^*}(f_{\mathsf{CSO}}) \\ &\leq \gamma_C\cdot\nu\cdot\mathcal{C}(f_{\mathsf{CSO}})/R \leq \gamma_C\cdot\nu\cdot\mathcal{C}(f_{\mathsf{OPT}})/R \\ &\leq \gamma_C\cdot\nu\cdot\mu\cdot\mathcal{T}(f_{\mathsf{OPT}})/R \\ &\leq \gamma_C\cdot\nu\cdot\mu\cdot\mathcal{M}(f_{\mathsf{OPT}}) \leq \gamma_C\cdot\nu\cdot\mu\cdot D,\end{aligned}$$

where p' is the flow-carrying path with the largest path delay in f_{CSO} and p^* is the flow-carrying path with the smallest path cost in f_{CSO}. Inequality (a) comes from Lem. 4. ■

Proof of Thm. 3.

Proof: According to [33, Lem. 2.6], non-decreasing, continuous functions must admit a feasible Nash-equilibrium flow. Therefore, f_{DNE} and f_{CNE} exist, and $|f_{\mathsf{DNE}}| = |f_{\mathsf{CNE}}| = R$.

As for the maximum delay of f_{DNE}, we have

$$\mathcal{M}(f_{\mathsf{DNE}}) \leq \alpha_D\cdot\mathcal{M}^* \leq \alpha_D\cdot\mathcal{M}(f_{\mathsf{OPT}}) \leq \alpha_D\cdot D.$$

As for the total cost of f_{DNE}, we have

$$\begin{aligned}\mathcal{C}(f_{\mathsf{DNE}}) &\overset{(a)}{\leq} \mu\cdot\mathcal{T}(f_{\mathsf{DNE}}) \overset{(b)}{\leq} \mu\cdot\alpha_D\cdot\mathcal{T}^* = \mu\cdot\alpha_D\cdot\mathcal{T}(f_{\mathsf{DSO}}) \\ &\leq \mu\cdot\alpha_D\cdot\mathcal{T}(f_{\mathsf{OPT}}) \overset{(c)}{\leq} \mu\cdot\nu\cdot\alpha_D\cdot\mathcal{C}(f_{\mathsf{OPT}}),\end{aligned}$$

where inequalities (a) and (c) comes from Lem. 6, and inequality (b) holds due to Lem. 5.

As for the total cost of f_{CNE}, we have

$$\mathcal{C}(f_{\mathsf{CNE}}) \le \alpha_C \cdot \mathcal{C}^* = \alpha_C \cdot \mathcal{C}(f_{\mathsf{CSO}}) \le \alpha_C \cdot \mathcal{C}(f_{\mathsf{OPT}}).$$

As for the maximum delay of f_{CNE}, we have

$$\begin{aligned}\mathcal{M}(f_{\mathsf{CNE}}) &= d^{p'}(f_{\mathsf{CNE}}) \le \nu \cdot c^{p'}(f_{\mathsf{CNE}}) \overset{(a)}{=} \nu \cdot \mathcal{C}(f_{\mathsf{CNE}})/R \\ &\le \frac{\nu \cdot \alpha_C \cdot \mathcal{C}(f_{\mathsf{CSO}})}{R} \le \frac{\mu \cdot \nu \cdot \alpha_C \cdot \mathcal{T}(f_{\mathsf{OPT}})}{R} \\ &\le \mu \cdot \nu \cdot \alpha_C \cdot \mathcal{M}(f_{\mathsf{OPT}}) \le \mu \cdot \nu \cdot \alpha_C \cdot D\end{aligned}$$

where the path p' is the flow-carrying path with the largest path delay in the flow f_{CNE}, and equality (a) comes from the definition of a cost-Nash-equilibrium flow. ■

Proof of Lem. 3.

Proof: It is easy to verify that with variables $\{x_e, \forall e \in E\}$, problem (7) can be casted as a convex program with a polynomial size. Hence f_r can be obtained in polynomial time.

We prove $\mathcal{M}(f_r) \le D$ by contradiction. Suppose $\mathcal{M}(f_r) > D$, namely there exists a $p' \in \mathcal{P}$ in f_r such that

$$d^{p'}(f_r) = \sum_{e:e\in p'} d_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}} \right) > D \text{ and } x^{p'} > 0. \quad (8)$$

Note that problem (7) can be formulated as follows

$$\min \sum_{p\in\mathcal{P}} \left[x^p \cdot \sum_{e:e\in p} c_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}} \right) \right] \quad (9a)$$

$$\sum_{p\in\mathcal{P}} x^p - \frac{1}{D} \cdot \sum_{p\in\mathcal{P}} \left[x^p \cdot \sum_{e:e\in p} d_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}} \right) \right] \ge r, \quad (9b)$$

$$x^p \ge 0, \quad \forall p \in \mathcal{P}. \quad (9c)$$

Different from the optimal solution f_r of the problem (7) (or the problem (9) equivalently), let's consider another flow f_r^* where we delete all flow rate $x^{p'}$ from p' in f_r, namely

$$x^p(f_r^*) = x^p(f_r) : \ \forall p \in \mathcal{P}\backslash\{p'\}; \quad x^p(f_r^*) = 0 : \ p = p',$$

where $x^p(f)$ is the flow rate assigned on the path p in f.

Since f_r is feasible to (9), we have

$$\sum_{p\in\mathcal{P}} x^p(f_r) - \frac{1}{D} \cdot \sum_{p\in\mathcal{P}} \left[x^p(f_r) \cdot \sum_{e:e\in p} d_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}}(f_r) \right) \right] \ge r,$$

implying that

$$\sum_{p\in\mathcal{P}} x^p(f_r) \cdot \left(D - \sum_{e:e\in p} d_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}}(f_r) \right) \right) \ge D \cdot r.$$

Hence it holds that

$$\begin{aligned}&\sum_{p\in\mathcal{P}} x^p(f_r) \cdot \left(D - \sum_{e:e\in p} d_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}}(f_r) \right) \right) \\ = &\sum_{p\in\mathcal{P}\backslash\{p'\}} x^p(f_r) \cdot \left(D - \sum_{e:e\in p} d_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}}(f_r) \right) \right) \\ &+ x^{p'}(f_r) \cdot \left(D - \sum_{e:e\in p'} d_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}}(f_r) \right) \right) \ge D \cdot r.\end{aligned}$$

Considering the inequality (8), we have

$$\sum_{p\in\mathcal{P}\backslash\{p'\}} x^p(f_r) \cdot \left(D - \sum_{e:e\in p} d_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}}(f_r) \right) \right) > D \cdot r.$$

Since the link delay function is non-decreasing, it holds that

$$d_e \left(\sum_{\bar{p}\in\mathcal{P}\backslash\{p'\}:e\in\bar{p}} x^{\bar{p}}(f_r) \right) \le d_e \left(\sum_{\bar{p}\in\mathcal{P}:e\in\bar{p}} x^{\bar{p}}(f_r) \right).$$

Therefore, we have

$$\begin{aligned}&\sum_{p\in\mathcal{P}} x^p(f_r^*) \cdot \left(D - \sum_{e:e\in p} d_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}}(f_r^*) \right) \right) \\ = &\sum_{p\in\mathcal{P}\backslash\{p'\}} x^p(f_r) \cdot \left(D - \sum_{e:e\in p} d_e \left(\sum_{\bar{p}\in\mathcal{P}\backslash\{p'\}:e\in\bar{p}} x^{\bar{p}}(f_r) \right) \right) \\ \ge &\sum_{p\in\mathcal{P}\backslash\{p'\}} x^p(f_r) \cdot \left(D - \sum_{e:e\in p} d_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}}(f_r) \right) \right) > D \cdot r.\end{aligned}$$

Hence, f_r^* is a feasible solution of the problem (9).

Comparing f_r^* with f_r in terms of the objective, we have

$$\begin{aligned}&\sum_{p\in\mathcal{P}} \left[x^p(f_r) \cdot \sum_{e:e\in p} c_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}}(f_r) \right) \right] \\ \overset{(a)}{>} &\sum_{p\in\mathcal{P}\backslash\{p'\}} \left[x^p(f_r) \cdot \sum_{e:e\in p} c_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}}(f_r) \right) \right] \\ \overset{(b)}{\ge} &\sum_{p\in\mathcal{P}\backslash\{p'\}} \left[x^p(f_r) \cdot \sum_{e:e\in p} c_e \left(\sum_{\bar{p}\in\mathcal{P}\backslash\{p'\}:e\in\bar{p}} x^{\bar{p}}(f_r) \right) \right] \\ = &\sum_{p\in\mathcal{P}} \left[x^p(f_r^*) \cdot \sum_{e:e\in p} c_e \left(\sum_{\bar{p}:e\in\bar{p}} x^{\bar{p}}(f_r^*) \right) \right],\end{aligned}$$

where the inequality in (a) is true since $x^{p'} > 0$ and the link cost function $c_e(x_e)$ is strictly positive for any $x_e > 0$. The inequality in (b) is true due to the non-decreasing property of the link cost function. Now we see that the feasible solution f_r^* has a total cost smaller than that of f_r, which is a contradiction with the assumption that f_r is the optimal solution of the problem (9). Thus, it must hold that $\mathcal{M}(f_r) \le D$. ■

Proof of Thm. 6.

Proof: Under the condition of $|f_{r^*}| = R$, considering that f_{DSO} minimizes total delay as well as $|f_{\mathsf{DSO}}| = R$, we have

$$|f_{\mathsf{DSO}}| - \mathcal{T}(f_{\mathsf{DSO}})/D \ge |f_{r^*}| - \mathcal{T}(f_{r^*})/D \ge r^*,$$

implying that f_{DSO} is feasible to the problem formulated in (7) with $r = r^*$. Therefore, we have $\mathcal{C}(f_{r^*}) \le \mathcal{C}(f_{\mathsf{DSO}})$.

According to Thm. 2, it holds that $\mathcal{C}(f_{\mathsf{DSO}}) \le \mu \cdot \nu \cdot \mathcal{C}(f_{\mathsf{OPT}})$. We hence have $\mathcal{C}(f_{r^*}) \le \mu \cdot \nu \cdot \mathcal{C}(f_{\mathsf{OPT}})$. ■

REFERENCES

[1] K. Kar, S. Sarkar, and L. Tassiulas, “Optimization based rate control for multipath sessions,” in *Teletraffic Science and Engineering*, 2001, vol. 4, pp. 805–816.

[2] J. Wang, L. Li, S. H. Low, and J. C. Doyle, “Can shortest-path routing and tcp maximize utility,” in *IEEE International Conference on Computer Communications*, 2003.

[3] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, “The case for vm-based cloudlets in mobile computing,” *IEEE Pervasive Computing*, 2009.

[4] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, “Fog computing and its role in the internet of things,” in *ACM Workshop on Mobile Cloud Computing*, 2012.

[5] X. Ran, H. Chen, Z. Liu, and J. Chen, “Delivering deep learning to mobile devices via offloading,” in *ACM Workshop Virtual Reality and Augmented Reality Network*, 2017.

[6] F. P. Kelly, A. K. Maulloo, and D. K. 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.

[7] S. H. Low and D. E. Lapsley, “Optimization flow control. i. basic algorithm and convergence,” *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 861–874, 1999.

[8] J. B. Orlin, “A polynomial time primal network simplex algorithm for minimum cost flows,” *Mathematical Programming*, vol. 78, no. 2, pp. 109–129, 1997.

[9] K. Paparrizos, N. Samaras, and A. Sifaleras, “An exterior simplex type algorithm for the minimum cost network flow problem,” *Computers & Operations Research*, vol. 36, no. 4, pp. 1176–1190, 2009.

[10] R. Dewil, P. Vansteenwegen, D. Cattrysse, and D. Van Oudheusden, “A minimum cost network flow model for the maximum covering and patrol routing problem,” *European Journal of Operational Research*, vol. 247, no. 1, pp. 27–36, 2015.

[11] T. Roughgarden, “The maximum latency of selfish routing,” in *ACM-SIAM Symposium on Discrete Algorithms*, 2004.

[12] J. R. Correa, A. S. Schulz, and N. E. Stier-Moses, “Fast, fair, and efficient flows in networks,” *Operations Research*, vol. 55, no. 2, pp. 215–225, 2007.

[13] F. Devetak, J. Shin, T. Anjali, and S. Kapoor, “Minimizing path delay in multipath networks,” in *IEEE International Conference on Communications*, 2011.

[14] Q. Liu, L. Deng, H. Zeng, and M. Chen, “A tale of two metrics in network delay optimization,” in *IEEE International Conference on Computer Communications*, 2018.

[15] ——, “On the min-max-delay problem: Np-completeness, algorithm, and integrality gap,” in *IEEE Information Theory Workshop*, 2017.

[16] Q. Liu, H. Zeng, and M. Chen, “Minimizing age-of-information with throughput requirements in multi-path network communication,” in *ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2019.

[17] S. Misra, G. Xue, and D. Yang, “Polynomial time approximations for multi-path routing with bandwidth and delay constraints,” in *IEEE International Conference on Computer Communications*, 2009.

[18] W. Zhang, J. Tang, C. Wang, and S. De Soysa, “Reliable adaptive multipath provisioning with bandwidth and differential delay constraints,” in *IEEE International Conference on Computer Communications*, 2010.

[19] 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 Transactions on Networking*, vol. 25, no. 1, pp. 199–209, 2017.

[20] R. Yu, G. Xue, and X. Zhang, “Application provisioning in fog computing-enabled internet-of-things: A network perspective,” in *IEEE International Conference on Computer Communications*, 2018.

[21] “Openfog use cases - autonomous driving scenario,” 2018. [Online]. Available: https://www.openfogconsortium.org/

[22] 5GPPP, “5g and the factories of the future,” *White Paper*, 2015.

[23] M. P. Mills, “The cloud begins with coal,” in *Digital Power Group*, 2013. [Online]. Available: http://www.tech-pundit.com/wp-content/uploads/2013/07/Cloud_Begins_With_Coal.pdf

[24] L. Yu, T. Jiang, M. Sun, and Y. Zou, “Cost-aware resource allocation for fog-cloud computing systems,” *arXiv preprint arXiv:1701.07154*, 2017.

[25] P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav, “It’s not easy being green,” in *ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, 2012.

[26] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, “Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption,” *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1171–1181, 2016.

[27] L. Yu, T. Jiang, and Y. Zou, “Fog-assisted operational cost reduction for cloud data centers,” *IEEE Access*, vol. 5, pp. 13 578–13 586, 2017.

[28] B. Grimmer and S. Kapoor, “Nash equilibrium and the price of anarchy in priority based network routing,” in *IEEE International Conference on Communications*, 2016.

[29] L. Kleinrock, “Analytic and simulation methods in computer network design,” in *ACM Spring Joint Computer Conference*, 1970.

[30] L. R. Ford and D. R. Fulkerson, “Maximal flow through a network,” *Canadian Journal of Mathematics*, 1956.

[31] D. Gangadharan, O. Sokolsky, I. Lee, B. Kim, C.-W. Lin, and S. Shiraishi, “Bandwidth optimal data/service delivery for connected vehicles via edges,” in *IEEE International Conference on Cloud Computing*, 2018.

[32] M. Grant and S. Boyd, “CVX: Matlab software for disciplined convex programming, version 2.1,” 2014.

[33] T. Roughgarden and É. Tardos, “How bad is selfish routing?” *Journal of the ACM*, vol. 49, no. 2, pp. 236–259, 2002.