COIN: Cost-Efficient Traffic Engineering with Various Pricing Schemes in Clouds

Gongming Zhao[1,2] Jingzhou Wang[1,2] Hongli Xu[1,2] Zhuolong Yu[3] Chunming Qiao[4]
[1]School of Computer Science and Technology, University of Science and Technology of China
[2]Suzhou Institute for Advanced Research, University of Science and Technology of China
[3]Department of Computer Science, Johns Hopkins University
[4]Department of Computer Science and Engineering, University at Buffalo

***Abstract*—The rapid growth of cloud services has brought a significant increase in inter-datacenter traffic. To transfer data among geographically distributed datacenters, cloud providers need to purchase bandwidth from ISPs. The data transferring cost has become one of the major expenses for cloud providers. Therefore, it is essential for a cloud provider to carefully allocate inter-datacenter traffic among the ISPs' links to minimize the costs. Exiting solutions mainly focus on the situations where all links adopt the same pricing scheme. However, in practice, ISPs usually provide multiple pricing schemes for their links due to market competition, which makes the existing solutions non-optimal. Thus, a new traffic engineering approach that considers various pricing schemes is needed. This paper presents COIN, a new framework for cost-efficient traffic engineering with various pricing schemes. We propose a partition rounding traffic engineering algorithm based on linear independence analysis. The approximation factors and time complexity are formally analyzed. We further conduct large-scale simulations with real-world topologies and datasets. Extensive simulation results show that COIN can save the data transferring cost by up to 54.54% compared with the state-of-the-art solutions.**

***Index Terms*—Traffic Engineering, Cost-Efficient, Inter-Datacenter Traffic, Clouds**

I. Introduction

Thanks to the convenience and efficiency of cloud computing, an increasing number of enterprises are moving their workloads to clouds [1]. Cloud providers usually host their services and workloads with large-scale datacenters [2]. To provide better availability and scalability, these datacenters are usually geographically distributed around the world [3]. In fact, the traffic between these datacenters (namely *inter-datacenter traffic*) is essential for world-wide deployed applications and services [4] [5]. Online applications such as storage, web search and social networks [6] keep generating a large volume of inter-datacenter traffic. A recent survey [7] highlights that nearly 70% of cloud providers have massive inter-datacenter traffic of more than 330TB per month.

To transfer data across datacenters, cloud providers need to purchase bandwidth from the Internet Service Provider (ISP) for the links they use [4]. The rapidly growing inter-datacenter traffic has led to a heavy cost on cloud providers [8] [9]. Considering that the ISPs provide multiple links with different service level agreements (SLA) for data transferring [10], it is essential for a cloud provider to carefully allocate inter-datacenter traffic among these inter-datacenter links, to minimize costs while satisfying each flow's QoS demand.

Nowadays, ISPs tend to provide a variety of pricing schemes for inter-datacenter traffic due to market competition [10]. In general, existing mainstream pricing schemes are determined by two components:

- *Billable traffic.* ISPs sample the inter-datacenter traffic and measure a few traffic metrics [11]. The inter-datacenter traffic is billed based on the measured traffic metrics. For example, with the *average (AVG) billable traffic*, the traffic is billed based on the average usage during the charging period. Similarly, ISPs also supports the *maximum (MAX) billable traffic*, the *percentile-based billable traffic* [12], *etc.*.
- *Billing method.* ISPs calculate the traffic cost based on the billable traffic and its billing methods. Representative billing methods include the *fixed billing method* [11], the *elastic billing method* [13] and the *usage-based billing method* [12]. For instance, under the fixed billing method, the cloud provider is allocated with a committed bandwidth capacity and needs to pay a fixed price.

In practice, ISPs can define their own *billable traffic* and *billing method*. Therefore, the traffic pricing schemes in real-production environment are diverse and flexible, which will be discussed in detail in Section III-B.

Several traffic engineering solutions [4] [11] [13] [14] [15] have been proposed to save inter-datacenter traffic transferring costs. For example, the authors in [11] present a scheduling algorithm based on the Committed Information Rate (CIR), which is a typical fixed billing method with the AVG billable traffic. The work [13] proposes a pricing scheme under the elastic billing method and the MAX billable traffic. Recent works [4] [14] [15] have designed efficient traffic engineering algorithms for the usage-based billing method with the percentile-based billable traffic to minimize costs of inter-datacenter traffic.

However, these existing works usually focus on the situation where all the inter-datacenter links adopt the same pricing scheme. They overlook the fact that ISPs usually provide various pricing schemes due to market competition [10]. That is to say, links configured with different pricing schemes could coexist between datacenters [16] [17]. According to

[15], the cost-optimization traffic engineering solution for a single pricing scheme may be extremely costly for others.

In this paper, we present COIN, a cost-efficient traffic engineering framework with various pricing schemes. In fact, the problem is very theoretically and mathematically complex. Even if we only consider the single pricing scheme adopting 95th-percentile billable traffic, it has been proven to be NP-complete [12] [15]. Nevertheless, we will consider various pricing schemes in our model, including nearly all current mainstream pricing schemes. It is obvious that our problem would be much more practical but more difficult compared with existing works. In general, our contributions can be highlighted as follows:

1) We comprehensively analyze the current pricing schemes for real-world inter-datacenter traffic and present COIN, the traffic engineering framework with various pricing schemes. To the best of our knowledge, it is the first traffic engineering framework that can efficiently deal with various pricing schemes.
2) We also propose an approximation algorithm based on the cost-efficient framework for three representative pricing schemes as a case study. Furthermore, we analyze its time complexity according to Gaussian elimination and discuss its applicability for different cases.
3) We conduct large-scale simulations using real-world topologies and datasets to show that COIN can achieve superior performance, *e.g.*, saving costs by up to 54.54% compared with the state-of-the-art solutions.

The rest of this paper is organized as follows. Section II presents the related works of inter-datacenter traffic engineering and strengthens our motivation. Section III introduces the preliminaries of our work, describes our traffic engineering framework, and presents a case study. We propose our algorithm in Section IV. The simulations are presented in Section V. We conclude this paper in Section VI.

II. Related Works

Current online applications incur a significant amount of inter-datacenter traffic for cloud provides [10], severely affecting network performance and resulting in huge financial expenses [4]. Therefore, efficient traffic engineering solutions are urgently needed. The existing traffic engineering works can be divided into two main categories: promoting network performance and saving traffic costs.

To promote network performance, many works are based on the classical traffic engineering protocol, *e.g.*, Border Gateway Protocol (BGP), which is used to distribute routing information among domains [18]. Several works [18] [19] [20] [21] focus on how to improve an aspect of network performance, *e.g.*, resiliency, load-balancing, stability, *etc.* Secci *et al.* [18] propose a model that achieves load-balancing for large inter-domain traffic volumes using game theory. Compared with BGP, this traffic engineering framework can achieve a much higher degree of resiliency and stability. The work [19] focuses on the safety problem associated with traffic engineering. The authors argue that traditional BGP is very vulnerable to a variety of attacks due to the lack of secure means of verifying the authenticity and legitimacy of traffic. Thus, the authors propose an authorization and authentication system that addresses the security problems associated with BGP. In short, BGP-based traffic engineering solutions are performance-aware but overlook the inter-datacenter traffic transferring costs charged by ISPs.

Apart from the network performance issues, the financial expenses charged by ISPs have also drawn attention from both academia and industries. The classical work [12] presents the optimal traffic engineering algorithms based on the AVG/MAX billable traffic and the usage based billing method. They further consider the situation where ISPs provide CIR pricing scheme (introduced in Section I), and provide the optimal traffic engineering based on the greedy algorithm. Li *et al.* [10] consider the cost-minimizing traffic engineering with the AVG billable traffic and the usage based billing method. In this work, the authors highlight that the unit costs of links are different since ISPs provide links with different capacities, and propose an algorithm based on alternating direction method of multipliers (ADMM) [22]. Entact [23] optimizes linear billable traffic, *e.g.*, AVG and MAX, while guaranteeing user experience. Overall, the billable traffic adopted in these works are all linear. Thus, designing traffic engineering solutions is relatively simple. Non-linear billable traffic [14], *e.g.*, percentile-based, is not taken into account due to its difficulty. The NP-completeness proof of finding optimal traffic engineering solution for 95th-percentile billable traffic can be found in [12].

Currently, to attract more cloud providers, many ISPs prevent penalties for occasional traffic surges [24] by adopting the 95th-percentile billable traffic. An increasing number of works [4] [14] [15] [25] [26] have explored how to save cost under this pricing scheme. The authors in [4] propose an online scheduler called *TrafficShaper* to schedule inter-datacenter traffic without prior knowledge of traffic arrivals. The work [25] mainly deals with inter-datacenter traffic in the Amazon EC2 cloud for video streaming providers. Singh *et al.* [14] present CASCARA for the pricing schemes adopting 95th-percentile billable traffic, which is a cloud traffic engineering framework to optimize inter-datacenter traffic allocation. They also prove the performance-awareness of CASCARA, by showing the bounded impact on links availability and latency. Based on CASCARA, the work [15] presents a traffic allocation scheme based on 95th-percentile billable traffic for practical CDNs considering flow allocation granularity and deviation of flows. The authors in [26] design a traffic engineering framework called *Pretium*, considering the dynamic prices of 95th-percentile billable traffic for inter-datacenter links. By implementing in a large WAN, Pretium reduces up to 80% costs. However, these works only consider that the inter-datacenter links all adopt the pricing scheme based on 95th-percentile billable traffic. Thus, how to design a general traffic engineering framework with various pricing schemes to save costs is an urgent and practical problem.

III. Preliminaries

A. Inter-datacenter Model

In this section, we first demonstrate the inter-datacenter traffic engineering model. Let $E = \{e_1, e_2, ..., e_{|E|}\}$ denote the set of inter-datacenter links owned by ISPs. Each link e has a physical bandwidth capacity $c(e)$. We use $F = \{f_1, f_2, ..., f_{|F|}\}$ to denote the set of flows that need to be transferred over inter-datacenter links. To satisfy the SLA requirement, we explore a suitable set of links E_f for each flow f. Similar to [14], we consider traffic engineering under the discrete-time mode, where a charging period (*e.g.*, a month) is divided into T time slots. Each time slot $t \in \{1, 2, ..., T\}$ has the same duration (*e.g.*, 5 minutes) [14] [15]. For instance, during a month of charging period, there are $30 \times 24 \times 12 = 8640$ 5-minute time slots.

Let variables $x_{t,e}^f$ denote the traffic amount of flow f through link e at time slot t. $k_{t,e}$ represents the total traffic amount over link e at time slot t, *i.e.*, $k_{t,e} = \sum_{f \in F} x_{t,e}^f$. The traffic demand of flow f at time slot t is denoted as d_t^f. Furthermore, we define the vector $\mathbf{k_e} = (k_{1,e}, k_{2,e}, ..., k_{T,e})$, to record the traffic amount over link e through all the time slots during a charging period.

B. Pricing Schemes

In practice, there are multiple links between two datacenters. Each link is configured with one pricing scheme, determined by the ISP [10]. The set of pricing schemes is denoted as $\mathcal{P}$. Specifically, a pricing scheme $p \in \mathcal{P}$ consists of two elements: billable traffic and billing method. We use a tuple to represent a pricing scheme, *i.e.*, $p = (b, m)$, where b denotes the billable traffic and m represents the billing method. In short, for each link e, ISPs will calculate a value based on the vector $\mathbf{k_e}$ as the billable traffic [12], denoted as $b(\mathbf{k_e})$. The billing method m is a function of billable traffic $b(\mathbf{k_e})$. Thus, the cost of link e can be expressed as $p(e) = m(b(\mathbf{k_e}))$. We give the detailed formulation of billable traffic and billing methods.

Billable traffic mainly includes three kinds: the average (AVG), the maximum (MAX) and the 95th-percentile [12].

AVG billable traffic calculates the average value of traffic during a charging period. Let $b_1(\mathbf{k_e})$ denote the average billable traffic of link e. It follows:

$$b_1(\mathbf{k_e}) = \frac{\sum_t k_{t,e}}{T}, \forall e \tag{1}$$

MAX billable traffic records the maximum traffic within the charging period, denoted as $b_2(\mathbf{k_e})$. We have:

$$b_2(\mathbf{k_e}) \geq k_{t,e}, \forall e, t \tag{2}$$

95th-percentile billable traffic first requires us to sort the traffic amount of all time slots in the ascending order. Then, 95th-percentile of the order is taken as the billable traffic. We use binary variable $\lambda_{t,e}$ to denote whether link e is charged in time slot t ($\lambda_{t,e} = 0$) or not ($\lambda_{t,e} = 1$). Similar to [14], we have:

$$b_3(\mathbf{k_e}) = \min z_e$$

$$S.t. \begin{cases} \sum_t \lambda_{t,e} = \lfloor \frac{T}{20} \rfloor, & \forall e \\ z_e \geq k_{t,e} - L\lambda_{t,e}, & \forall e, t \end{cases} \tag{3}$$

where L is a large integer constant. The first equation means that free time slots only take 5%. The second equation calculates the value of 95th-percentile billable traffic, *i.e.*, the minimum value of z_e.

Billing method describes how the ISPs charge cloud providers according to billable traffic. Current mainstream billing methods include the following three kinds.

Fixed billing method means that an ISP provides a committed bandwidth $c_1(e)$ for link e, and the inter-datacenter traffic amount should not exceed the bandwidth capacity. The ISP would charge a fixed price $s_1(e)$ [12]. We have:

$$\begin{cases} m_1(b_i(\mathbf{k_e})) = s_1(e), & \forall e \\ b_i(\mathbf{k_e}) \leq c_1(e), & \forall e \end{cases} \tag{4}$$

Elastic billing method provides a bandwidth threshold $c_2(e)$ and charges a fixed price $s_2(e)$. If the traffic volume is beyond the threshold, extra cost will be charged [13] and the unit price is denoted as $q_2(e)$. We use $b_i'(\mathbf{k_e})$ to represent the extra traffic on link e for charging, Thus we have:

$$\begin{cases} m_2(b_i(\mathbf{k_e})) = s_2(e) + b_i'(\mathbf{k_e}) \cdot q_2(e), & \forall e \\ b_i'(\mathbf{k_e}) = \max[0, b_i(\mathbf{k_e}) - c_2(e)], & \forall e \end{cases} \tag{5}$$

Usage based billing method is also a widely adopted pricing scheme. The cloud provider needs to pay the ISP according to the unit cost ($ per Mbps) and the billable traffic (Mbps). We assume the unit cost for link e is $q_3(e)$. It follows:

$$m_3(b_i(\mathbf{k_e})) = b_i(\mathbf{k_e}) \cdot q_3(e), \forall e \tag{6}$$

It is worth noting that the specific *charging prices*, *e.g.*, $s_1(e)$ in Eq. (4), $s_2(e)$ and $q_2(e)$ in Eq. (5), $q_3(e)$ in Eq. (6), are also important parameters determined by ISPs. We will show that how different charging prices affect the cost results in our simulations.

We demonstrate the formulation of a pricing scheme as an example. Specifically, we assume the pricing scheme is based on the AVG billable traffic and the usage based billing method, *i.e.*, $p = (b_1, m_3)$ and $p(e) = m_3(b_1(\mathbf{k_e}))$. Combining Eqs. (1) and (6), it follows:

$$\begin{cases} b_1(\mathbf{k_e}) = \frac{\sum_t k_{t,e}}{T}, & \forall e \\ p(e) = b_1(\mathbf{k_e}) \cdot q_3(e), & \forall e \end{cases} \tag{7}$$

In all, ISPs need to first determine billable traffic, and then charge the billable traffic according to the billing method. Since we introduce 3 kinds of billable traffic and 3 kinds of billing methods in this section, there are a total of 9 pricing schemes. Considering the diversity of pricing schemes, it is necessary and practical to design a general framework with various pricing schemes.

C. Cost-Efficient Traffic Engineering Framework

In this section, we give the formulation of the cost-efficient traffic engineering framework with various pricing schemes. We focus on the traffic engineering between two datacenters for simplicity, since the traffic between two arbitrary datacenters is independent of others [13]. We should note

that our algorithm can be easily extended to deal with traffic engineering among multiple datacenters. Recall that the set of inter-datacenter links is denoted as E and the set of pricing schemes is denoted as $\mathcal{P}$. We use E_p to denote the set of links with pricing scheme p. The traffic engineering decision is determined by calculating $x_{t,e}^f$, meaning how much traffic should flow f allocate to link e at time slot t.

When allocating traffic, we should consider the following constraints similar to [27]: 1) *Link Capacity Constraint:* The total traffic amount on every link at any time slot should not exceed its capacity. 2) *Traffic Demand Constraint:* The traffic demand of every flow should be satisfied at any time slot. In addition, to meet the SLA requirement, flow f can only be allocated to the links belonging E_f. We formulate the traffic engineering framework as follows:

$$\min \sum_{e \in E} p(e)$$
$$S.t. \begin{cases} k_{t,e} = \sum_{f \in F} x_{t,e}^f, & \forall e, t \\ k_{t,e} \leq c(e), & \forall e, t \\ \sum_{e \in E_f} x_{t,e}^f = d_t^f, & \forall f, t \\ p(e) = m(b(\mathbf{k_e})), & \forall e \in E_p \\ x_{t,e}^f \geq 0, & \forall e, f, t \end{cases} \tag{8}$$

The first set of equations calculates the traffic volume on link e at time slot t. The second set of inequalities represents the link capacity constraint, that on every link e at any time slot t, the traffic volume $k_{t,e}$ should not exceed the capacity $c(e)$. The third set of equations describes the traffic demand constraint for any inter-datacenter flow f at any time slot t. In this equation, flow f can only be allocated to link $e \in E_f$, due to the SLA requirement. The fourth set of equations expresses the cost of link e under pricing scheme p. Our objective is to minimize the total cost of all links charged by the ISP, *i.e.*, $\sum_{e \in E} p(e)$. Based on the above framework, we can combine it with specific pricing schemes to acquire a cost-efficient traffic engineering solution.

D. A Case Study

Pricing Schemes	Billable Traffic	Billing Method
p_1	AVG (b_1)	fixed (m_1)
p_2	MAX (b_2)	elastic (m_2)
p_3	95th-percentile (b_3)	usage based (m_3)

TABLE I: Representative Pricing Schemes

In this section, we will show how to use the general traffic engineering framework with specific pricing schemes by a case study. In this case, we choose three representative and widely adopted pricing schemes [11] [13] [14], as shown in Table I. The first pricing scheme p_1 consists of the AVG billable traffic and the fixed billing method. The second pricing scheme p_2 is determined by the MAX billable traffic and the elastic billing method. The third pricing scheme is based on the 95th-percentile billable traffic and the usage based billing method. The pricing schemes set $\mathcal{P} = \{p_1, p_2, p_3\}$. For each flow f, we need to determine how to allocate the traffic to the links configured with three pricing schemes to minimize the total cost. Combining the general framework in Eq. (8) and three pricing schemes shown in Section III-B, we give the following formulation:

$$\min \sum_{e \in E_{p_1}} p_1(e) + \sum_{e \in E_{p_2}} p_2(e) + \sum_{e \in E_{p_3}} p_3(e)$$
$$S.t. \begin{cases} k_{t,e} = \sum_{f \in F} x_{t,e}^f, & \forall e, t \\ k_{t,e} \leq c(e), & \forall e, t \\ \sum_{e \in E_f} x_{t,e}^f = d_t^f, & \forall t, f \\ b_1(\mathbf{k_e}) = \frac{\sum_t k_{t,e}}{T}, & \forall e \in E_{p_1} \\ b_1(\mathbf{k_e}) \leq c_1(e), & \forall e \in E_{p_1} \\ p_1(e) = s_1(e), & \forall e \in E_{p_1} \\ b_2(\mathbf{k_e}) \geq k_{t,e}, & \forall e \in E_{p_2}, t \\ b_2'(\mathbf{k_e}) = \max[0, b_2(\mathbf{k_e}) - c_2(e)], & \forall e \in E_{p_2} \\ p_2(e) = s_2(e) + b_2'(\mathbf{k_e}) \cdot q_2(e), & \forall e \in E_{p_2} \\ \sum_t \lambda_{t,e} = \lfloor \frac{T}{20} \rfloor, & \forall e \in E_{p_3} \\ z_e \geq k_{t,e} - M\lambda_{t,e}, & \forall e \in E_{p_3}, t \\ b_3(\mathbf{k_e}) = \min z_e, & \forall e \in E_{p_3} \\ p_3(e) = b_3(\mathbf{k_e}) \cdot q_3(e), & \forall e \in E_{p_3} \\ \lambda_{t,e} \in \{0, 1\}, & \forall e \in E_{p_3}, t \\ x_{t,e}^f \geq 0, & \forall e, f, t \end{cases} \tag{9}$$

In this formulation, the former three equations/inequalities are the same as those in Eq. (8). The fourth to the sixth equations/inequalities calculate the cost according to pricing scheme p_1, *i.e.*, Eqs. (1) and (4). The seventh to the ninth equations/inequalities denote the pricing scheme p_2, *i.e.*, Eqs. (2) and (5). The tenth to the thirteenth equations/inequalities imply the pricing scheme p_3, *i.e.*, Eqs. (3) and (6). Our objective is to minimize the cost of traffic under the three various pricing schemes, *i.e.*, $\sum_{e \in E_{p_1}} p(e) + \sum_{e \in E_{p_2}} p(e) + \sum_{e \in E_{p_3}} p(e)$.

Since the formulation consists of binary variables, it is a 0-1 integer programming problem [28], which is computationally hard to solve. Moreover, according to [12], finding the optimal solution for the 95th-percentile pricing scheme is NP-complete. Thus, how to design an efficient algorithm for Eq. (9) is a tricky problem.

IV. Algorithm Design

A. Algorithm Description

As shown in our case study defined in Eq. (9), for each link $e \in E_{p_3}$, the sum of binary variables is $\lfloor \frac{T}{20} \rfloor$, *i.e.*, $\sum_t \lambda_{t,e} = \lfloor \frac{T}{20} \rfloor, \forall e \in E_{p_3}$. The classical solution for 0-1 integer programming problem, *e.g.*, the randomized rounding algorithm [29], only works for the situation where $\lfloor \frac{T}{20} \rfloor = 1$. However, we should note that $\lfloor \frac{T}{20} \rfloor > 1$ in the most of practical cases, which means the infeasibility of the classical randomized rounding algorithm for our problem. Thus, we present a new approximation algorithm, called Partition Rounding-based Traffic Engineering (PTE), to solve the problem.

Algorithm 1 PTE: Partition Rounding-based Traffic Engineering

Step1: Solve the relaxed formulation
Construct a linear program by replacing the integral constraints $\lambda_{t,e} \in \{0,1\}$ with $\lambda_{t,e} \in [0,1]$
Obtain the optimal solutions $\{\widehat{\lambda}_{t,e}\}$ and $\{\widehat{x}^f_{t,e}\}$
Step2: Acquire a set of feasible solution
for each $e \in E_{p_3}$ **do**
    Create partitions and construct a set of linear indeterminate equations according to Eq. (10)
    Solve the set of linear indeterminate equations and obtain a set of feasible solutions $\widehat{P}(C)$
    Choose one partition C with probability of $\widehat{P}(C)$
    Derive integral solution $\{\widetilde{\lambda}_{t,e}\}$ by setting all the variables within the chosen partition to one, and others are rounded to zero
Return $\{\widetilde{\lambda}_{t,e}\}$ and $\{\widehat{x}^f_{t,e}\}$

Our algorithm consists of two main steps and is formally described in Alg. 1. In the first step, we construct the 0-1 integer programming according to Eq. (9) and relax the binary variables $\lambda \in \{0,1\}$ into linear ones, *i.e.*, $\lambda \in [0,1]$. Then we apply a linear program solver (*e.g.*, CPLEX [30]) to acquire the fractional solutions, denoted as $\{\widehat{x}^f_{t,e}\}$ and $\{\widehat{\lambda}_{t,e}\}$.

In the second step, we determine how to choose free time slots for each link $e \in E_{p_3}$, *i.e.*, obtaining integer solutions $\{\widetilde{\lambda}_{t,e}\}$. Note that, the following process is executed for each link $e \in E_{p_3}$ independently, and we use N to represent $\lfloor \frac{T}{20} \rfloor$ for simplicity. The key idea is to create partitions for variables and calculate the probability of choosing each partition. Specifically, we first divide these variables into several partitions and each partition contains N distinct variables. Obviously, there are $\binom{T}{N}$ different partitions. We use variable $P(C)$ to denote the probability that the partition C is chosen. Then, we construct the set of linear indeterminate equations as follows:

$$\begin{cases} \sum_C P(C) = 1 \\ \sum_{C:\lambda_{t,e} \in C} P(C) = \widehat{\lambda}_{t,e}, \quad \forall t \end{cases} \tag{10}$$

It contains $T+1$ equations and $\binom{T}{N}$ variables. Note that, since $\binom{T}{N} = O(T^{\min\{N,T-N\}})$, the number of variables could be very huge. To this end, we will show a method to reduce the number of variables from $\binom{T}{N}$ down to T based on the linear independence analysis, and prove the time complexity of solving Eq. (10) is only $O(T^3)$ based on Gaussian elimination (see Section IV-C for details). Then, the set of equations can be easily solved by an equation solver (*e.g.*, SymPy [31]) to acquire a set of feasible solutions, *i.e.*, $\widehat{P}(C)$. Similar to the randomized rounding algorithm, we choose one partition C with the probability of $\widehat{P}(C)$. Note that, all the variables in the chosen partition will be rounded to one, and others will be rounded to zero. In this way, we can acquire integral solutions, *i.e.*, $\{\widetilde{\lambda}_{t,e}\}$.

B. Performance Analysis

We first prove the correctness of our PTE algorithm.

Theorem 1: The proposed PTE algorithm can guarantee that for each link $e \in E_{p_3}$, the free time slots exactly take 5% of the whole charging period, *i.e.*, the constraint $\sum_t \lambda_{t,e} = \lfloor \frac{T}{20} \rfloor$ in Eq. (9) is satisfied.

Proof: In the second step of PTE, for each link $e \in E_{p_3}$, we only choose one partition according to the solution of Eq. (10), and all the variables in the partition will be rounded to one, while others are rounded to zero. Considering that each partition contains N variables, the constraint $\sum_t \lambda_{t,e} = \lfloor \frac{T}{20} \rfloor = N$ is satisfied. It means that for each link $e \in E_{p_3}$, the free time slots take exactly 5% of all time slots. ■

We then give two famous lemmas for probability analysis.

Lemma 2: (Chernoff Bound [32]) Given n independent variables: $x_1, x_2, ..., x_n$, where $\forall x_i \in [0,1]$. Let $\mu = \mathbb{E}[\sum_{i=1}^n x_i]$. Then, we have $\mathbf{Pr}[\sum_{i=1}^n x_i \geq (1+\epsilon)\mu] \leq e^{\frac{-\epsilon^2\mu}{2+\epsilon}}$ and $\mathbf{Pr}[\sum_{i=1}^n x_i \leq (1-\epsilon)\mu] \leq e^{\frac{-\epsilon^2\mu}{2}}$, where ϵ is an arbitrarily positive value.

Lemma 3: (Union Bound [33]) Given an accountable set of n events: $A_1, A_2, ...A_n$, each event A_i happens with probability $\mathbf{Pr}(A_i)$. Then, $\mathbf{Pr}(A_1 \cup A_2 \cup ... \cup A_n) \leq \sum_{i=1}^n \mathbf{Pr}(A_i)$.

Next, we prove the following lemma and analyze the approximation factor.

Lemma 4: The proposed algorithm can guarantee $E[\lambda_{t,e}] = \widehat{\lambda}_{t,e}$, $\forall e \in E_{p_3}$, t.

Proof: According to Eq. (10), for each link $e \in E_{p_3}$ and each time slot t, variable $\lambda_{t,e}$ corresponds to an equation:

$$\sum_{C:\lambda_{t,e} \in C} P(C) = \widehat{\lambda}_{t,e} \tag{11}$$

We choose the partition C with the probability of $\widehat{P}(C)$ and all the variables in the chosen partition will be rounded to one. Assuming that $\lambda_{t,e}$ is in partitions $C_1, C_2, ..., C_k$, according to Eq. (11), we have $\sum_{i=1}^k P(C_i) = \widehat{\lambda}_{t,e}$. The probability of choosing the partitions containing $\lambda_{t,e}$ is exactly $\widehat{\lambda}_{t,e}$. Therefore, we have $E[\lambda_{t,e}] = 1 \times \widehat{\lambda}_{t,e} + 0 \times (1 - \widehat{\lambda}_{t,e}) = \widehat{\lambda}_{t,e}$. ■

Theorem 5: PTE acquires a feasible solution with an approximation factor of $O(\log |F|)$, where $|F|$ is the number of flows.

Proof: As discussed in Lemma 4, all the binary variables can keep the expectation values, *i.e.*, $E[\lambda_{t,e}] = \widehat{\lambda}_{t,e}, \forall e \in E_{p_3}$. We define $z_e = \min[k_{t,e} - M\lambda_{t,e}], \forall e \in E_{p_3}$. It follows $\widehat{z}_e = \min[k_{t,e} - M\widehat{\lambda}_{t,e}], \forall e \in E_{p_3}$. For each link $e \in E_{p_3}$, we have $E[z_e] = E[\min[k_{t,e} - M\lambda_{t,e}]] = \min[k_{t,e} - M\widehat{\lambda}_{t,e}] = \widehat{z}_e$. Similarly, we have $E[b_3(\mathbf{k}_e)] = E[\min z_e] = \min \widehat{z}_e = \widehat{b}_3(\mathbf{k_e})$ and $E[p_3(e)] = E[b_3(\mathbf{k}_e) q_3(e)] = \widehat{b}_3(\mathbf{k_e}) q_e(e) = \widehat{p}_3(e)$, for each link $e \in E_{p_3}$.

We define the constant $p_{max} = \max\{p_3(e), \forall e \in E_{p_3}\}$. According to Lemma 2, we have:

$$\begin{aligned} &\mathbf{Pr}[\frac{p_3(e)}{p_{max}} \geq (1+\epsilon) E[\frac{p_3(e)}{p_{max}}]] \leq e^{\frac{-\epsilon^2\mu}{2+\epsilon}} \\ \Rightarrow &\mathbf{Pr}[\frac{p_3(e)}{\widehat{p}_3(e)} \geq (1+\epsilon)] \leq e^{\frac{-\epsilon^2\mu}{2+\epsilon}} \leq \frac{1}{|F|^2} \end{aligned} \tag{12}$$

Here $\mu = \frac{p_3(e)}{p_{max}}$. We can know that $\frac{1}{|F|^2} \to 0$ when the number of flows grows. The solution to Eq. (12) follows:

$$\epsilon \geq \frac{2\log|F| + \sqrt{4\log^2|F| + 16\mu\log|F|}}{2\mu},$$
$$\Rightarrow \epsilon \geq \frac{2\log|F|}{\mu} + 2 = O(\log|F|), \quad |F| \geq 2 \tag{13}$$

Thus, we have:

$$p_e < O(\log|F|)\widehat{p}_3(e), \forall e \in E_{p_3} \tag{14}$$

Let $|E_{p_3}|$ denote the number of links $e \in E_{p_3}$. Combining Lemma 3, *i.e.*, Union Bound, it follows:

$$\begin{aligned}&\mathbf{Pr}[\bigcup_{e\in E_{p_3}} \frac{p_3(e)}{\widehat{p}_3(e)} \geq (1+\epsilon)] \\ &\leq \sum_{e\in E_{p_3}} \mathbf{Pr}[\frac{p_3(e)}{\widehat{p}_3(e)} \geq (1+\epsilon)] \leq \frac{|E_{p_3}|}{|F|^2}\end{aligned} \tag{15}$$

When the number of flows grows, $\frac{|E_{p_3}|}{|F|^2} \to 0$. It concludes that the total costs of link $e \in E_{p_3}$ will not exceed the optimal results by a factor of $O(\log|F|)$ with a high probability.

Since the value of variables $\widehat{x}^f_{t,e}$ derived by solving the LP, *i.e.*, the first step of PTE, will not be affected by partition rounding in the second step of PTE, it means for all time slots t and all link e, the value of $\widehat{k}_{t,e} = \sum_f \widehat{x}^f_{t,e}$ derived by solving the LP remains the same. Thus, the billable traffic will not be changed, and after rounding procedure, we still have $p_1(e) = \widehat{p}_1(e), \forall e \in E_{p_1}$ and $p_2(e) = \widehat{p}_2(e), \forall e \in E_{p_2}$, where $p_1(e)$ and $p_2(e)$ represent the values derived by PTE and $\widehat{p}_1(e)$ and $\widehat{p}_2(e)$ denote the values derived by solving the LP. Therefore, the total cost denoted by $\mathbb{P}$ derived by the PTE algorithm follows:

$$\begin{aligned}\mathbb{P} &= \sum_{e\in E_{p_1}} p_1(e) + \sum_{e\in E_{p_2}} p_2(e) + \sum_{e\in E_{p_3}} p_3(e) \\ &< \sum_{e\in E_{p_1}} \widehat{p}_1(e) + \sum_{e\in E_{p_2}} \widehat{p}_2(e) + \sum_{e\in E_{p_3}} O(\log|F|)\widehat{p}_3(e)\end{aligned} \tag{16}$$

Compared with the optimal solution $\widehat{\mathbb{P}}$ by solving the LP, we finally have:

$$\begin{aligned}\frac{\mathbb{P}}{\widehat{\mathbb{P}}} &= \frac{\sum_{i=1,2}\sum_{e\in E_{p_i}} p_i(e) + \sum_{e\in E_{p_3}} O(\log|F|)\widehat{p}_3(e)}{\sum_{i=1,2,3}\sum_{e\in E_{p_i}} \widehat{p}_i(e)} \\ &= O(\log|F|)\end{aligned} \tag{17}$$

It concludes that our algorithm can achieve the approximation factor of $O(\log|F|)$ for the solution. ■

C. Analysis on Linear Indeterminate Equations

Compared with the classical randomized rounding solution, our proposed algorithm contains one more step: solving the set of linear indeterminate equations in Eq. (10). In this section, we first prove that Eq. (10) is solvable. Then, we analyze the time complexity of solving Eq. (10). Recall that the original formulation requires $\sum_t \lambda_{t,e} = N, \forall e \in E_{p_3}$, where $N = \lfloor \frac{T}{20} \rfloor$. There are T variables for each $e \in E_{p_3}$.

Theorem 6: The set of linear indeterminate equations in Eq. (10) is solvable.

Proof: Even though we have $T+1$ equations in Eq. (10), we first show that there are actually T linear independent equations. Since each partition C contains N different variables, every $P(C)$ will appear in exact N equations in Eq. (10) besides $\sum_C P(C) = 1$. Thus, if we add all the equations except $\sum_C P(C) = 1$, we can acquire $N \cdot \sum_C P(C) = \sum_t \widehat{\lambda}_{t,e} = N$. This is equivalent to $\sum_C P(C) = 1$. Thus, this equation is actually redundant.

Then, since $T \leq \binom{T}{N}$, the number of equations is not more than that of variables. According to [34], the set of linear indeterminate equations is solvable. ■

In our problem setting, during a month, we consider a five-minute interval. Therefore, we have $30 \times 24 \times 12 = 8640$ time intervals, *i.e.*, $T = 8640$, and $N = \lfloor \frac{T}{20} \rfloor = 432$. It brings us a very intractable problem with a very huge number of variables, *i.e.*, $\binom{8640}{432} = O(10^{743})$, far beyond the capacity of modern computers. In fact, we can use the Gaussian elimination algorithm to minimize the difficulty and substantially decrease the time complexity.

Theorem 7: The time complexity of solving the set of linear indeterminate equations in Eq. (10) is $O(T^3)$.

Proof: We discuss this theorem according to the value of N. If $N = T - 1$, we can directly use the Gaussian elimination algorithm and the time complexity is $O(T^3)$ [35]. Due to limited space, we omit the detailed analysis in this paper. Otherwise, we only need to create a part of $P(C)$ for calculation. The rest of $P(C)$ are set to 0 directly. Specifically, in order to avoid linear dependence, we create partitions by the following steps. We assume for a link $e \in E_{p_3}$, $C = \{\lambda_{a_1,e}, \lambda_{a_2,e}, ..., \lambda_{a_N,e}\}$, and we always have $a_1 < a_2 < ... < a_N$. The initial setting is $a_1 = 1, a_2 = 2, ..., a_N = N$. We consider the number $(a_1 a_2 ... a_N)$ under base-T+1 positional notation. Then, we gradually add one to the last number a_N until there are T partitions. Therefore, the new set of linear equations only contains T variables. The time complexity of solving equations is still $O(T^3)$. Furthermore, since all the coefficients in Eq. (10) are zero or one, the calculation would be much simpler compared with general sets of linear indeterminate equations. ■

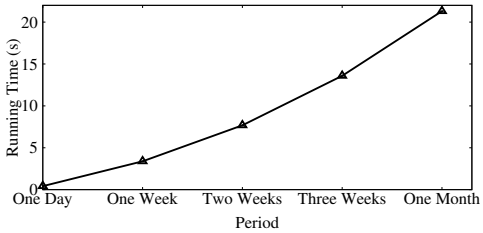


Fig. 1: The Running Time of Solving the Linear Indeterminate Equations for Different Charging Periods

To enhance persuasiveness, we conduct experiments to evaluate the running time of solving Eq. (10) for different charging periods. For instance, if the charging period is one month, the set of equations has $30 \times 24 \times 12 = 8640$ variables. The experiments are conducted with an i7-11800H CPU and

32GB RAM. The equation solver is SymPy and the results are shown in Fig. 1.

The results show that the running time is negligible compared with the charging period. For instance, when the charging period is one week, the duration of solving equations only takes 3.38s. The evaluation results fully demonstrate that solving the set of equations is applicable in practice. We should note that, with an advanced algorithm for linear equations [36] or a high-performance computer, the running time could be reduced substantially.

D. Discussion on Case Study

In our case study defined in Section III-D, we only consider three pricing schemes. However, we should note that, our algorithm can still work in other scenarios.

Scenario 1: There does not exist any pricing scheme adopting 95th-percentile billable traffic. Then the case study becomes a linear programming problem. We can use a standard LP solver to acquire the optimal solutions directly, *i.e.*, the second step of PTE will not be executed.

Scenario 2: There are multiple pricing schemes adopting 95th-percentile billable traffic. For instance, there exist two pricing schemes both adopting 95th-percentile billable traffic, one of which uses the elastic billing method and the other of which uses the fixed billing method. We can use our algorithm to first acquire a set of optimal but fractional solutions, *i.e.*, $\widehat{\lambda}_{t,e}$. For each link with different pricing scheme adopting 95th-percentile billable traffic, we execute the second step of PTE independently. Finally, we acquire a set of feasible solutions with bounded approximation factors.

Scenario 3: There exist other kinds of percentile based billable traffic. In the work [4], authors consider qth-percentile billable traffic where q ranges from 50-100. In this case, we only need to change the equation in Eq. (9), *i.e.*, $\sum_t \lambda_{t,e} = \lfloor \frac{T}{20} \rfloor, \forall e$ into $\sum_t \lambda_{t,e} = \lfloor (1 - \frac{q}{100})T \rfloor, \forall e$. Then, we can similarly use the PTE algorithm to derive a solution.

These discussions fully demonstrate the applicability of our algorithm. Under different scenarios, we can easily modify our algorithm to acquire cost-efficient traffic engineering solutions with bounded approximation factors.

V. Performance Evaluation

A. Simulation Settings

1) Benchmarks: We use the following three algorithms as benchmarks: (1) Load-balancing (**LB**) [37] is a widely adopted traffic allocation scheme. It always chooses the link with the least load ratio for flows. (2) **CASCARA** [14] is the latest research on inter-datacenter traffic with the 95th-percentile billable traffic and the usage based billing method. It directly computes the formulation defined in Eqs. (3) and (6) to acquire the cost-effective allocation results. (3) Global Integral Assignment (**GIA**) [38] schedules flow through an integral assignment manner, *i.e.*, assigning the entire flow to the same link greedily. In our cases, it greedily schedules flows to the least cost link.

2) Flow Datasets: We use the power law for the flow-size distribution, where 20% of all flows account for 80% of traffic volume [39]. Specifically, we set: (1) **Dataset I** simulates the traffic from datacenters of a search engine [4]. The average traffic is set to 1 Mbps. (2) **Dataset II** simulates the traffic from datacenters of a video streaming provider [25]. The average traffic is set to 1.5Mbps. Along with time, the traffic volume of flows in two datasets will change randomly. Specifically, 50% of flows will enlarge traffic size by at most 2 times, and 50% of flows will reduce traffic size by at most 100%. By default, there are 5000 inter-datacenter flows.

3) Charging Prices: To quantitatively describe the financial expenses, we need to determine the specific cost of links. We set charging prices according to the Amazon EC2 cloud [10]. In our simulation, we mainly consider two kinds of charging prices, denoted as A and B. For clear description, we have listed the parameter settings in Table II. For instance, with charging prices B, given the fixed billing method, the cloud provider needs to pay $\$40$ per month for 25Gbps bandwidth.

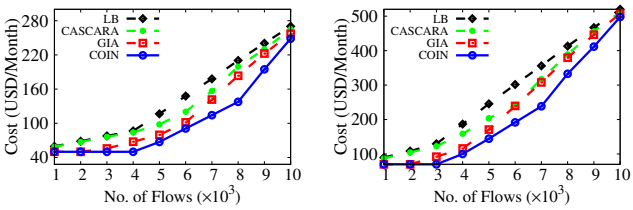
Methods	Formulations	Parameters	Prices	
			A	B
Fixed	Eq. (4)	$s_1(e)$ ($)	30	40
		$c_1(e)$ (Gbps)	30	25
Elastic	Eq. (5)	$s_2(e)$ ($)	20	30
		$c_2(e)$ (Gbps)	20	15
		$q_2(e)$ ($/Gbps)	5	8
Usage based	Eq. (6)	$q_3(e)$ ($/Gbps)	2	4

TABLE II: Charging Prices of Billing Methods

4) Inter-datacenter Settings: According to [10] [25], we set the number of links between two datacenters to 3 by default. The pricing schemes are set to the same as those in our case study as shown in Table I, *i.e.*, p_1, p_2, p_3. The physical bandwidth capacity of each link is set to 50Gbps [10]. We consider one month charging period and the time interval is set to 5-minute [15] by default.

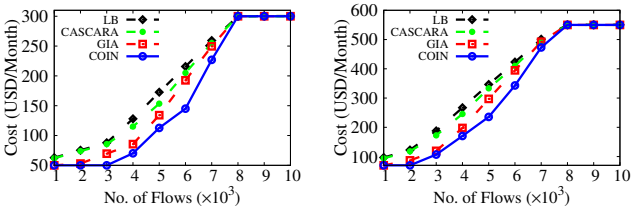
B. Simulation Results

We run four groups of simulations to check the cost efficiency of COIN. The first group of experiments shows the total costs by varying the number of flows between two datacenters. The results are shown in Figs. 2-3. From the figures, we can learn that overall, the cost results derived by four algorithms increase with a growing number of flows. Specially, with dataset II, when the number of flows exceeds 7×10^3, the cost results remain the same, as shown in Fig. 3. That is because all links are fully loaded, and there is no space for cost optimization. From the figures, we can learn that COIN always acquires the lowest costs compared with other benchmarks. For instance, in Fig. 2(a), given 8×10^3 flows, the cost results of LB, CASCARA, GIA and COIN are $\$210.27$, $\$199.43$, $\$183.42$ and $\$137.58$ per month, respectively, with dataset I and charging prices A. COIN can reduce costs by 34.76%, 31.15% and 25.13%



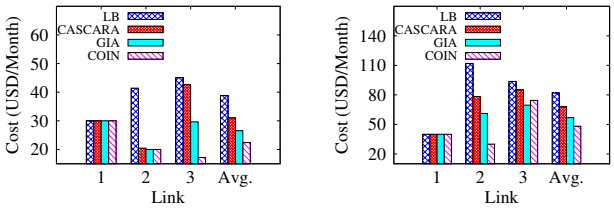
(a) Charging Prices A (b) Charging Prices B

Fig. 2: Cost vs. Number of Flows with the Dataset I



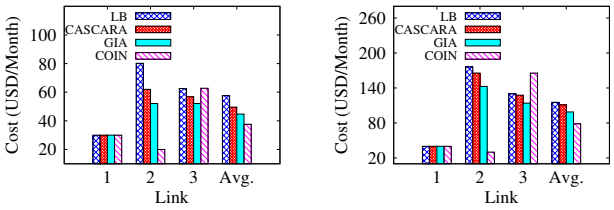
(a) Charging Prices A (b) Charging Prices B

Fig. 3: Cost vs. Number of Flows with the Dataset II



(a) Charging Prices A (b) Charging Prices B

Fig. 4: Cost of Each Link with the Dataset I



(a) Charging Prices A (b) Charging Prices B

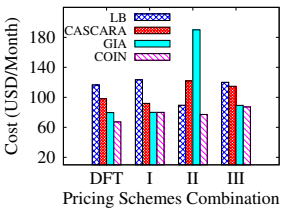
Fig. 5: Cost of Each Link with the Dataset II

compared with LB, CASCARA and GIA, respectively. The reason why LB achieves the highest costs is that it does not take pricing schemes into consideration in traffic allocation. In addition, since CASCARA only optimizes costs for the 95th-percentile sampling method and usage based billing method, it does not perform well when there are other pricing schemes. Since GIA chooses links with the fixed and the elastic billing methods first, it overlooks the optimal solutions which adopt the link with usage based billing method. COIN, however, will generally consider all links with different pricing schemes to calculate the cost-optimal traffic engineering solutions. Thus, the cost results derived by COIN are always the lowest among all the algorithms. Moreover, we can learn from Fig. 3(b), given the dataset II and charging prices B, COIN can reduce the cost by 24.55%, 21.42% and 11.44% on average, compared with LB, CASCARA and GIA, respectively.

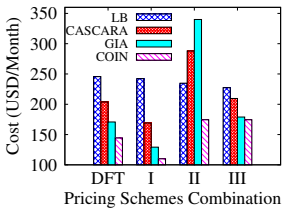
The second group of simulations shows the specific cost of each link given 5×10^3 flows. The results are shown in Figs. 4-5. From the figures, we can see that since link 1 is configured with pricing scheme p_1 adopting the fixed billing method, the cost results of four algorithms are always the same. For instance, in Fig. 4(a), the costs of link 1 are all \$30 per month, with charging prices A. It is worth noting that COIN cannot assure that all links achieve the lowest cost. For instance, in Figs. 5(a)-5(b), the costs of link 3 derived by COIN are slightly higher than those of GIA. This is because GIA will greedily allocate traffic to link 2 with elastic billing method first, rather than link 3 with the usage based billing method. However, on average, COIN still achieves the lowest costs. For instance, in Fig. 5(b), given dataset II and charging prices B, the average costs derived by LB, CASCARA, GIA and COIN are \$115.41, \$111.05, \$98.88 and \$78.50 per month, respectively. COIN reduces costs by 31.98%, 29.27% and 20.61% compared with LB, CASCARA and GIA, respectively,

To further prove that COIN is able to perform well under different scenarios, the third group of simulations will compare the cost results derived by four algorithms when ISP provides different pricing schemes combinations for three links. In our case study, the settings of **Combination Default (DFT)** are three links each with pricing scheme p_1, p_2, p_3, respectively. We show the cost results when the ISP provides the following pricing schemes combination for three links. **Combination I**: three links are with p_1, p_1, p_2, respectively. **II**: three links are with p_2, p_2, p_3, respectively. **III**: three links are with p_3, p_3, p_1, respectively. The results are shown in Figs. 6-7. We can observe that COIN always achieves the minimum cost among the four algorithms, no matter what combination is given. For instance, in Fig. 6(b), when ISP provides combination I with charging prices B, COIN can reduce costs by 54.54%, 34.91% and 14.72% compared with LB, CASCARA and GIA, respectively. It is worth noting that in Figs. 7(a)-7(b), when given combination II, we can see that the cost results of CASCARA and GIA are extremely high. This is because CASCARA is designed for the 95th-percentile sampling methods, and cannot perform well when there are two links using the MAX sampling method. In addition, GIA will greedily allocate traffic to the links with the elastic billing method, acquiring a very costly result. Since COIN is designed for the general situation, in Fig. 7(a), it achieves the minimum cost and reduces costs by 28.65%, 43.26% and 52.34% compared with LB, CASCARA and GIA, respectively, with dataset II and charging prices A. Moreover, when ISP provides combination III shown in Fig. 7(a), COIN reduces costs by 14.38%, 8.96% and 8.33% compared with LB, CASCARA and GIA, respectively.

In the fourth group of simulations, we will compare the cost results by varying the number of links. We also consider all pricing schemes in this simulation, as shown in Table III.

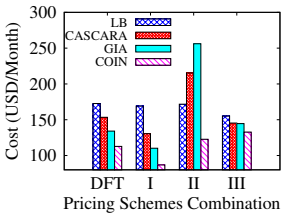


(a) Charging Prices A

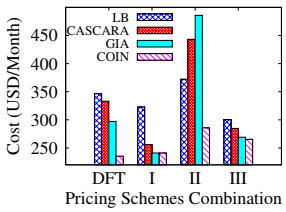


(b) Charging Prices B

Fig. 6: Cost of Each Pricing Schemes Combination for Three Links with the Dataset I

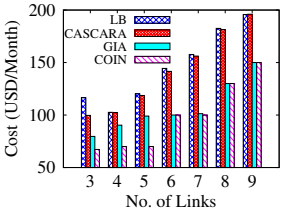


(a) Charging Prices A

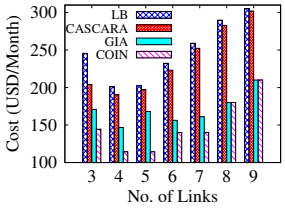


(b) Charging Prices B

Fig. 7: Cost of Each Pricing Schemes Combination for Three Links with the Dataset II

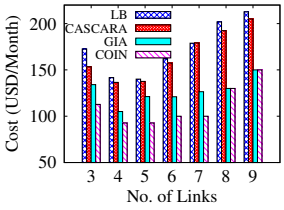


(a) Charging Prices A

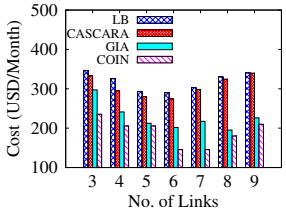


(b) Charging Prices B

Fig. 8: Cost vs. Number of Links with the Dataset I



(a) Charging Prices A



(b) Charging Prices B

Fig. 9: Cost vs. Number of Links with the Dataset II

Pricing Schemes	Billable Traffic	Billing Method
p_4	AVG (b_1)	elastic (m_2)
p_5	AVG (b_1)	usage based (m_3)
p_6	MAX (b_2)	fixed (m_1)
p_7	MAX (b_2)	usage based (m_3)
p_8	95th-percentile (b_3)	fixed (m_1)
p_9	95th-percentile (b_3)	elastic (m_2)

TABLE III: Pricing Schemes

Specifically, when there are N links, the pricing scheme of each link is $p_1, p_2, ..., p_N$, respectively. The performance comparisons are shown in Figs. 8-9. It is obvious that no matter how many links are provided, COIN can always acquire the lowest cost results. For instance, in Fig. 8(a), when the ISP provides 5 links with charging prices A, the cost results derived by LB, CASCARA, GIA and COIN are $\$120.22, \$118.63, \$99.03$ and $\$70$, respectively. COIN reduces costs by 41.67%, 40.68% and 29.29% compared with LB, CASCARA and GIA, respectively. It is worth noting that when given 8 links or 9 links, GIA can acquire nearly the same results as COIN does. This is because there exist multiple links with the fixed billing method, and GIA will greedily allocate traffic to these links. Meanwhile, these links are capable of holding all the traffic. Therefore, choosing the links with the fixed billing method happens to be the optimal solution. However, we should note that, when given other number of links, GIA may be much more costly than COIN does. For instance, in Fig. 9(b) with dataset II and charging prices B, when there are 6 links, COIN reduces costs by 49.82%, 47.08% and 27.86% compared with LB, CASCARA and GIA, respectively. It is also worth noting that the total costs may decrease when given more links. For instance, in Figs. 8-9, when there are 4 links, the cost results are all lower than those of 3 links. This is because the fourth link is with elastic billing method, and within the bandwidth threshold, it is cheaper than usage based billing method.

In conclusion, these simulations fully demonstrate that COIN can reduce inter-datacenter traffic costs by up to 54.54% compared with other benchmarks. Moreover, COIN is able to work well under different scenarios not only when given different number of flows (Figs. 2-3), different pricing schemes combinations (Figs. 4-7), or different number of links (Figs. 8-9), but also when given different flow datasets (I and II) or different charging prices (A and B), since COIN is a general framework, considering all pricing schemes.

VI. Conclusion

In this paper, we have presented COIN, a new cost-efficient traffic engineering framework supporting various pricing schemes. We also propose an approximation algorithm based on the cost-efficient framework for three representative pricing schemes as a case study. Furthermore, we analyze its time complexity and discuss its applicability for different cases. Extensive simulations have been conducted to show the superior performance of our proposed algorithm.

Acknowledgement

The corresponding author of this paper is Hongli Xu. This article was supported in part by the National Science Foundation of China (NSFC) under Grant 62102392; and in part by the National Science Foundation of Jiangsu Province under Grant BK20210121; and in part by Anhui Initiative in Quantum Information Technologies under Grant AHY150300; and in part by Hefei Municipal Natural Science Foundation No. 2022013.

REFERENCES

[1] J. Wang, G. Zhao, H. Xu, H. Huang, L. Luo, and Y. Yang, "Robust service mapping in multi-tenant clouds," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, 2021, pp. 1–10.

[2] E. Cayirci, "Modeling and simulation as a cloud service: a survey," in *2013 Winter Simulations Conference (WSC)*. IEEE, 2013, pp. 389–400.

[3] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," pp. 68–73, 2008.

[4] W. Li, X. Zhou, K. Li, H. Qi, and D. Guo, "Trafficshaper: Shaping inter-datacenter traffic to reduce the transmission cost," *IEEE/ACM Transactions on Networking*, vol. 26, no. 3, pp. 1193–1206, 2018.

[5] H. Xu and B. Li, "Joint request mapping and response routing for geo-distributed cloud services," in *2013 Proceedings IEEE INFOCOM*. IEEE, 2013, pp. 854–862.

[6] M. Noormohammadpour and C. S. Raghavendra, "Datacenter traffic control: Understanding techniques and tradeoffs," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 2, pp. 1492–1525, 2017.

[7] F. Research, "The future of data center wide-area networking," http://www.forrester.com, JUN. 2022.

[8] F. M. Puspita, K. Seman, B. M. Taib, and Z. Shafii, "Improved models of internet charging scheme of single bottleneck link in multi qos networks," *Open Access*, 2013.

[9] S. Bodamer, "Charging in multi-service networks," *Internal Report of University of Stuttgart*, no. 29, 1998.

[10] W. Li, K. Li, D. Guo, G. Min, H. Qi, and J. Zhang, "Cost-minimizing bandwidth guarantee for inter-datacenter traffic," *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 483–494, 2016.

[11] A. D. M. Africa and F. M. Dimaala, "Enhanced traffic scheduling algorithm by implementing committed information rate (cir) in broadband systems for a disaster-stricken footprint in telecommunications," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 4, p. 4209, 2020.

[12] M. Adler, R. K. Sitaraman, and H. Venkataramani, "Algorithms for optimizing the bandwidth cost of content delivery," *Computer Networks*, vol. 55, no. 18, pp. 4007–4020, 2011.

[13] M. Motiwala, A. Dhamdhere, N. Feamster, and A. Lakhina, "Towards a cost model for network traffic," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 1, pp. 54–60, 2012.

[14] R. Singh, S. Agarwal, M. Calder, and P. Bahl, "Cost-effective cloud edge traffic engineering with cascara," in *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*, 2021, pp. 201–216.

[15] H. Chen, H. Zhan, H. Tan, H. Xu, W. Shan, S. Chen, and X.-Y. Li, "Online traffic allocation based on percentile charging for practical cdns," in *2022 IEEE/ACM 30th International Symposium on Quality of Service (IWQOS)*, 2022, pp. 1–10.

[16] "Other methods for metering bandwidth usage," https://www.stackscale.com/blog/95-percentile-metering-billing-bandwidth/, JUN. 2022.

[17] V. Valancius, C. Lumezanu, N. Feamster, R. Johari, and V. V. Vazirani, "How many tiers? pricing in the internet transit market," in *Proceedings of the ACM SIGCOMM 2011 Conference*, 2011, pp. 194–205.

[18] S. Secci, K. Liu, and B. Jabbari, "Efficient inter-domain traffic engineering with transit-edge hierarchical routing," *Computer Networks*, vol. 57, no. 4, pp. 976–989, 2013.

[19] S. Kent, C. Lynn, and K. Seo, "Secure border gateway protocol (s-bgp)," *IEEE Journal on Selected areas in Communications*, vol. 18, no. 4, pp. 582–592, 2000.

[20] B. Schlinker, H. Kim, T. Cui, E. Katz-Bassett, H. V. Madhyastha, I. Cunha, J. Quinn, S. Hasan, P. Lapukhov, and H. Zeng, "Engineering egress with edge fabric: Steering oceans of content to the world," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, 2017, pp. 418–431.

[21] K.-K. Yap, M. Motiwala, J. Rahe, S. Padgett, M. Holliman, G. Baldus, M. Hines, T. Kim, A. Narayanan, A. Jain *et al.*, "Taking the edge off with espresso: Scale, reliability and programmability for global internet peering," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, 2017, pp. 432–445.

[22] Y. Wang, W. Yin, and J. Zeng, "Global convergence of admm in nonconvex nonsmooth optimization," *Journal of Scientific Computing*, vol. 78, no. 1, pp. 29–63, 2019.

[23] Z. Zhang, M. Zhang, A. G. Greenberg, Y. C. Hu, R. Mahajan, and B. Christian, "Optimizing cost and performance in online service provider networks." in *NSDI*, 2010, pp. 33–48.

[24] N. W.B, "Transit tactic-gaming the 95th percentile," https://drpeering.net/, Apr. 2015.

[25] Y. Feng, B. Li, and B. Li, "Jetway: Minimizing costs on inter-datacenter video traffic," in *Proceedings of the 20th ACM international conference on Multimedia*, 2012, pp. 259–268.

[26] V. Jalaparti, I. Bliznets, S. Kandula, B. Lucier, and I. Menache, "Dynamic pricing and traffic engineering for timely inter-datacenter transfers," in *Proceedings of the 2016 ACM SIGCOMM Conference*, 2016, pp. 73–86.

[27] J. Wang, G. Zhao, H. Xu, Y. Zhao, X. Yang, and H. Huang, "Trust: Real-time request updating with elastic resource provisioning in clouds," in *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*, 2022, pp. 620–629.

[28] E. L. Johnson, M. M. Kostreva, and U. H. Suhl, "Solving 0-1 integer programming problems arising from large scale planning models," *Operations Research*, vol. 33, no. 4, pp. 803–819, 1985.

[29] P. Raghavan and C. D. Tompson, "Randomized rounding: a technique for provably good algorithms and algorithmic proofs," *Combinatorica*, vol. 7, no. 4, pp. 365–374, 1987.

[30] "cplex," https://www.ibm.com/analytics/cplex-optimizer, JUN. 2022.

[31] "Sympy," https://github.com/sympy/sympy, JUN. 2022.

[32] M. Hellman and J. Raviv, "Probability of error, equivocation, and the chernoff bound," *IEEE Transactions on Information Theory*, vol. 16, no. 4, pp. 368–372, 1970.

[33] C. Tellambura, "Evaluation of the exact union bound for trellis-coded modulations over fading channels," *IEEE transactions on communications*, vol. 44, no. 12, pp. 1693–1699, 1996.

[34] H. J. S. Smith, "Xv. on systems of linear indeterminate equations and congruences," *Philosophical transactions of the royal society of london*, no. 151, pp. 293–326, 1861.

[35] N. J. Higham, "Gaussian elimination," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 3, no. 3, pp. 230–238, 2011.

[36] G. Shi, B. D. Anderson, and U. Helmke, "Network flows that solve linear equations," *IEEE Transactions on Automatic Control*, vol. 62, no. 6, pp. 2659–2674, 2016.

[37] S. Afzal and G. Kavitha, "Load balancing in cloud computing–a hierarchical taxonomical classification," *Journal of Cloud Computing*, vol. 8, no. 1, pp. 1–24, 2019.

[38] L. Zhang, Z. Li, C. Wu, and M. Chen, "Online algorithms for uploading deferrable big data to the cloud," in *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*. IEEE, 2014, pp. 2022–2030.

[39] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, "The nature of data center traffic: measurements & analysis," in *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement*, 2009, pp. 202–208.