A Case for Pricing Bandwidth: Sharing Datacenter Networks With Cost Dominant Fairness

Li Chen, Yuan Feng, Baochun Li, *Fellow, IEEE*, and Bo Li, *Fellow, IEEE*

Abstract—Unlike other resources such as CPU or memory in a virtual machine, inter-virtual-machine (inter-VM) bandwidth has not been explicitly priced in datacenter networks. In this article, we argue that tenants of an IaaS cloud computing platform should be given the flexibility to pay more for explicitly priced datacenter bandwidth beyond traditional virtual machines, in order to achieve better (or more predictable) application performance. We show that a much simpler design principle can be followed to allocate bandwidth fairly, and desirable properties related to fairness can be more easily achieved, compared with state-of-the-art proposals. We call such a design principle *cost dominant fairness*, which stipulates that bandwidth should be allocated based on the total cost that a tenant incurs for running its applications in the cloud. Guided by the principle of *cost dominant fairness*, we explore the design space of pricing inter-VM bandwidth, as well as achieving fair bandwidth sharing among multiple tenants. Through our study, we believe that it is best to assign per-VM-pair weights based on individualized prices. We present a distributed bandwidth allocation algorithm that is theoretically supported by a network utility maximization formulation, and practically implemented as a shim layer at each virtual machine. We are also concerned with practical issues of billing, where discounts are needed to ensure that a tenant only pays for the bandwidth share that it is allocated. Finally, we have evaluated our pricing framework and per-VM-pair weighted fair bandwidth allocation in the Mininet emulation testbed and simulations.

Index Terms—Datacenter networks, bandwidth allocation, fairness, pricing

1 Introduction

The essence of Infrastructure-as-a-Service(IaaS) is to allow multiple tenants to share resources such as CPU cycles, storage, and bandwidth. When tenants pay for running applications in the cloud, CPU, memory and storage resources are explicitly priced in today's datacenters, in the form of virtual machine (VM) pricing. Yet, inter-VM bandwidth in datacenter networks has *not* been explicitly priced. As a result, inter-VM bandwidth is typically shared in a *best-effort* fashion, which makes it challenging for both cloud providers and tenants to reason about how bandwidth is allocated as a scarce resource.

It has been well understood that the performance of a cloud application heavily depends on its allocated share of inter-VM bandwidth. With the best-effort way of sharing bandwidth, multiple tenants may compete for bandwidth on a shared bottleneck link, and such competition leads to congestion and degraded performance for all. On the other hand, a tenant may wish to pay more for better or a predictable level of performance, e.g., with respect to task finishing times in MapReduce jobs [1], [2], [3]. Leaving bandwidth free of charge to the tenants, together with the best-effort fashion of sharing bandwidth, have implied that a tenant is *not* able to pay more to attain better performance by obtaining a larger or reserved share of bandwidth. Largely due to their competition for bandwidth, multiple tenants interfere with one another in a shared datacenter, with unpredictable costs on reserving VM instances.

The tussle between free-of-charge bandwidth as a resource and the need to allocate bandwidth fairly across competing tenants in a datacenter has led to perplexing solutions. It is intuitive to allocate bandwidth across tenants based on how much they have paid for their cloud services; but since there is no charge for bandwidth, existing solutions in the literature proposed to allocate bandwidth based on the number of VMs that tenants have leased and paid for. With these solutions, a tenant who wishes to pay for a better share of bandwidth will need to lease more VMs. Effectively, the tenant uses a convoluted way to pay for more bandwidth by paying for more CPU cycles and memory instead, which it may not need.

In this paper, we believe that regardless of the number of VMs a tenant leases, it should be given the flexibility to obtain a better or more predictable share of inter-VM bandwidth, by *pricing bandwidth* explicitly, and *independently* from VM pricing. In order to achieve that, reserved bandwidth shares should be priced accordingly. In addition, most

- *Li Chen is with the School of Computing and Informatics, University of Louisiana at Lafayette, Lafayette, LA 70504 USA. E-mail: li.chen@louisiana.edu.*
- *Yuan Feng and Baochun Li are with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S, Canada. E-mail: yfeng@eecg.toronto.edu, bli@ece.toronto.edu.*
- *Bo Li is with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, China. E-mail: bli@cse.ust.hk.*

Manuscript received 12 Sept. 2020; revised 6 Dec. 2020; accepted 7 Dec. 2020. Date of publication 18 Dec. 2020; date of current version 7 Jan. 2021. (Corresponding author: Li Chen.) Recommended for acceptance by Y. Yang. Digital Object Identifier no. 10.1109/TPDS.2020.3045709

tenants may be content with paying more for a better *best-effort share* of bandwidth rather than paying for exclusive bandwidth reservations, and higher prices should reflect a better bandwidth share. While some may argue that tenants may be reluctant to pay more for bandwidth in addition to what they have paid for VMs, we believe that, rather than *having to* "play the lottery" with unpredictable performance due to interference, or to implicitly pay for more bandwidth by leasing more VMs that it does not need, tenants may prefer the *option* to pay more for better performance.

In the literature, it has been quite a challenge to allocating bandwidth *fairly* across competing tenants, since a few desirable properties need to be maintained, such as strategy-proofness, communication pattern independence, and symmetry [4]. A surprising but important advantage that tips the scale in favour of bandwidth pricing is that, if inter-VM bandwidth is priced explicitly, a much simpler design principle can be followed to allocate bandwidth fairly, and the aforementioned properties related to fairness can be more easily achieved. We call such a design principle *cost dominant fairness*, which stipulates that bandwidth should be allocated based on the total cost that a tenant incurs for running its applications in the cloud; and on the other hand, a tenant will pay proportionally more if its applications have enjoyed better performance.

With the objective of achieving the principle of *cost dominant fairness*, we explore the solution space of pricing inter-VM bandwidth, as well as achieving fair bandwidth sharing among multiple tenants. The original contributions that we present in this paper are as follows. *First*, we make a strong case for pricing both guaranteed and best-effort bandwidth shares, and propose the principle of cost dominant fairness (Section 3). *Second*, by exploring the solution space of bandwidth pricing strategies, we propose a personalized pricing framework for best-effort bandwidth (Section 4). *Third*, to allocate best-effort bandwidth according to cost dominant fairness, we propose to assign weights to each pair of VMs, determined based on personalized bandwidth prices that the tenants have chosen (Section 5). Based on the per-VM-pair weights assigned, we formulate the problem as a network utility maximization problem, where the objective is to achieve *weighted proportional fairness* among different flows with network capacity constraints (Section 6). *Fourth*, we consider the practical issues of billing, where discounts may need to be given to ensure that a tenant only pays for the bandwidth share that it is allocated (Section 7). *Finally*, we present a system-level design to practically implement our proposed bandwidth allocator as a shim layer at each virtual machine, which scales naturally and is topology-agnostic, and evaluate its performance in the Mininet emulation testbed as well as simulations with fat-tree topologies (Section 8).

2 Related Work

Making Bandwidth Reservations. The first and most intuitive solution to satisfy cloud tenants is to support *bandwidth reservations*, which are guaranteed to tenants ([5], [6], [7], [8], [9], *etc.*). Guo *et al.* proposed SecondNet with a new abstraction called virtual data centers (VDCs) [5], precisely to provide facilities to support such bandwidth guarantees between pairs of VMs. Rodrigues *et al.* proposed Gatekeeper [6], with the objective of reserving egress and ingress bandwidth at each VM. Such reservations give a tenant the illusion of a single, non-blocking switch connecting each of its VMs. Ballani *et al.* proposed Oktopus, with similar network abstractions to SecondNet and Gatekeeper [7]. By profiling the traffic pattern of several cloud applications, Xie *et al.* argued that cloud applications exhibit predictable time-varying traffic behaviour on the order of tens of seconds, and proposed a new fine-grained network reservation abstraction [8], so that the utilization of bandwidth can be improved by time-division multiplexing.

Sharing Best-Effort Bandwidth Fairly. If tenants do not need strict bandwidth guarantees to run their applications, the *fairness* of sharing best-effort bandwidth among multiple tenants then becomes a primary concern [3], [4], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20]. Shieh *et al.* proposed Seawall [3], which stipulated that on all network links, the share of bandwidth obtained by a VM serving as the traffic source is proportional to the VM's predefined network weight. Different from per-VM weights in Seawall, Lam *et al.* proposed NetShare [10], where weights were assigned to each tenant, and used to achieve weighted max-min fairness on congested links across multiple tenants. With different weight assignment mechanisms, Popa *et al.* pointed out that there is a fundamental tradeoff between network proportionality and high resource utilization [4]. They proposed a number of desirable properties with respect to best-effort sharing of inter-VM bandwidth, and discussed the pros and cons of existing bandwidth sharing policies. Three new bandwidth allocation policies have also been proposed, allowing the cloud provider to navigate the tradeoff space and to obtain the maximum sets of non-conflicting desirable properties. From a different perspective of sharing networks in a private datacenter, Chen *et al.* proposed performance-centric fairness [21], [22], [23], [24] and utility max-min fairness [25], [26], [27] to guide the problem of bandwidth allocation among flows from multiple data parallel applications. In this paper, the proposed notion of fairness combined with our pricing strategy regulates the allocation of both the guaranteed and the best-effort bandwidth, which differentiates us from the existing efforts.

Pricing Bandwidth. Most existing works in the literature assume that inter-VM bandwidth in a datacenter network is free of charge to tenants. Yet, it has been acknowledged that the VM occupancy time and its associated cost to a tenant are heavily influenced by its bandwidth share across VMs. To mitigate the problem of unpredictable tenant costs, Ballani *et al.* argued that the total cost to a tenant should be *independent* from the location of its VMs [28].[1] To achieve such location-independent costs, it then proposed that the cost to a tenant per unit time should be the greater of its VM occupancy and bandwidth costs, and that a baseline bandwidth should be guaranteed to each VM. While our proposed notion of cost dominant fairness share the general philosophy that application performance in a multi-tenant datacenter should proportionally reflect what a tenant pays for, we

1. TIVC [8] has also assumed that bandwidth costs are passed on to tenants, but it has not addressed the challenge of how bandwidth is to be priced explicitly. Instead, it referred the readers to [28].

wish to systematically explore the entire solution space and to propose the best possible solution that achieves cost dominant fairness, rather than settling with a preliminary strawman approach that requires bandwidth reservations.

Jalaparti *et al*. proposed a framework of dynamic pricing and traffic engineering for inter-datacenter network [29]. Their main objective is to achieve high social welfare, which does not enforce any fairness criterion. Numfabric [30] is also aimed at maximizing network utility through weighted fair queueing packet scheduling. Guo *et al*. proposed a bandwidth pricing strategy for intra-datacenter networks [31], with a special focus on bandwidth overcommitment to improve network utilization. Our proposed strategy does not overcommit, yet provide and enforce a unified pricing framework for both the guaranteed bandwidth and the best-effort shares.

Our Motivation. Although the above recent proposals are towards the direction of providing more predictable sharing on datacenter networks, unfortunately, none of them reveals a complete system design, be they either focus on virtual network abstraction or network sharing mechanism. As summarized in FairCloud, each one of them exhibits certain drawbacks in the tradeoff space. The reason, which we believe, is largely because they are restricted by the current pricing model used in the cloud service. As a consequence, we argue that it is highly desired to have a complete system design that supports the coexistence of both bandwidth guaranteed traffic and traffic sharing the network based on a suitable sharing policy. Users should be given full flexibility to choose from services with different qualities. Furthermore, the system should be not only a "mixture" of existing solutions, but should be a unified framework that is guided by a fundamental principle.

In this paper, we propose a differentiated service model in sharing the datacenter network. Our system is built based on a newly designed pricing framework that prices the bandwidth in different service types in a datacenter network on top of the traditional pricing model for VMs, following a fundamental fairness principle that we call "cost-dominant fairness." The key idea lies in that a cloud user will pay proportionally more if its applications have enjoyed better performance with more bandwidth resources.

The reason we propose a pricing model for bandwidth resources in the datacenter network is that, *first*, we believe pricing is necessary to a differentiated and hence predictable performance, which is the first-class objective in IaaS platforms. Without the pricing framework, fairness is an abstract concept that is hard to justify. What cloud users really desire is an inter-tenant isolated service with predicated quality. They would rather pay for the corresponding received service, just as they pay for different types of instances in the cloud, rather than care about the fairness among different users. In essence, the objective of choosing the cloud service is to get their own demand satisfied and pay for what they get, instead of comparing the bandwidth they receive from other users. *Second*, by having a pricing framework, many fundamental tradeoffs in sharing the datacenter network can be resolved naturally. Take the tradeoff between network proportionality and high utilization as an example. When pricing is involved, the emphasize on high utilization can be relaxed a bit. The reason is that as long as users pay for their usage, whether the bandwidth is fully utilized or not will not bother the cloud provider too much. Rational users will naturally try to minimize their usage to cut the cost. As a cloud provider, its only objective is to maximize its profit, which is the truth in reality.

3 A Case for Bandwidth Pricing

In this section, we first establish convincing arguments for explicit bandwidth pricing, and then introduce the notion of cost dominant fairness.

Guaranteed and Best-Effort Bandwidth. Before we present a strong case for pricing bandwidth explicitly, we first consider the requirements of cloud applications related to bandwidth. Consistent with most existing works, we believe that both *guaranteed* and *best-effort* bandwidth shares should *co-exist* in the same datacenter network, shared by multiple tenants.

Guaranteed bandwidth shares are reserved by making explicit bandwidth reservations, subject to availability and admission control [5], [6], [7], [8]. Such bandwidth guarantees are provided to a tenant irrespective of competition from other tenants sharing the same datacenter. In contrast, best-effort bandwidth corresponds to the traditional way of sharing bandwidth without any *a priori* reservation (as in the current Internet). Since guaranteed and best-effort bandwidth shares co-exist, best-effort traffic from different tenants will need to share the *residual* link bandwidth at each network link, after guaranteed bandwidth shares are provisioned and accounted for.

An intuitive rationale for supporting both guaranteed and best-effort bandwidth sharing is to cater to a variety of tenants with different needs. Bandwidth guarantees cater to the need of those tenants demanding explicit performance predictability. If an important MapReduce job needs to have a lower bound on its worst-case performance, it may request bandwidth guarantees for its data-intensive shuffle operations among its VMs. Best-effort shares of bandwidth, on the other hand, may be sufficient for tenants who lack precise predictions of bandwidth consumption, or those who have no strict performance expectations.

The Need to Charge for Reserved Bandwidth Shares. Now, let us consider a datacenter with both guaranteed and best-effort bandwidth sharing. Without explicitly pricing models, a tenant will naturally subscribe to the guaranteed bandwidth service. Since guaranteed bandwidth shares enjoy higher priorities than best-effort shares, the tenant will request a guaranteed bandwidth share that is more than what it needs at peak demand, even if it does not have strict needs for performance predictability. Even those tenants who actually require predictable performance would request larger shares of reserved bandwidth than their actual predicted demand "just in case," as idle bandwidth does not cost them extra. The result is catastrophic to the bottom line of an IaaS cloud provider's revenue: less residual bandwidth is available for best-effort use, while a large amount of reserved bandwidth remains unused. Naturally, an IaaS provider has to price and charge for bandwidth reservations explicitly.

A Case for Explicit and Differentiated Pricing on Best-Effort Bandwidth Shares. While most would intuitively agree that it

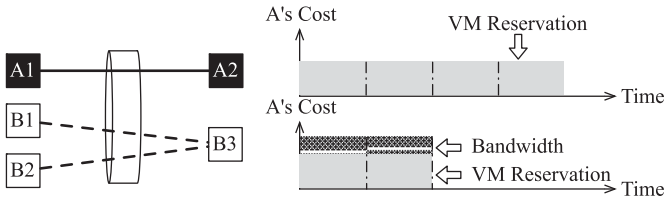


Fig. 1. A motivating example of pricing best-effort bandwidth shares.

is necessary to charge for guaranteed bandwidth shares, they may wonder why best-effort bandwidth shares should also be priced explicitly, and charged to the tenants.

The first observation we wish to make is that, residual best-effort bandwidth on each network link is a *scarce* resource, shared by competing flows from multiple tenants. Without explicit pricing for best-effort bandwidth, the general rule of thumb in existing papers is to allocate bandwidth based on the payment from competing tenants. For example, bandwidth can be allocated proportionally to the number of VMs that tenants have leased and paid for [4]. With these solutions, a tenant who wishes to be allocated a better share of bandwidth will need to lease more VMs. It is a contrived way to pay for more bandwidth by paying for more CPU cycles and memory instead, which the tenant may not need. If best-effort bandwidth is priced explicitly, it can be allocated proportionally to the bandwidth costs charged to the tenants, rather than the number of VMs. By pricing the tenant's bandwidth shares independently from its VMs, we can effectively decouple the pricing of bandwidth as a resource from that of CPU cycles and memory.

In the example shown in Fig. 1, two tenants share the datacenter network in a best-effort fashion. Tenant A initiated 2 identical VMs and B initiated 3 of the same type. By default, bandwidth is not priced, and we suppose that VMs from A and B will run for the same amount of time (4 time intervals), during which their inter-VM traffic (A1-A2, B1-B3 and B2-B3) is sharing a bottleneck link, as illustrated in the figure. In this case, tenant A is not able to get a larger bandwidth share than B, simply because with fewer VMs, A's total cost is smaller than B, and it will be unfair to B if A's share is larger. However, if bandwidth is priced independently from VMs, tenant A will have an opportunity to request for a larger share of best-effort bandwidth by paying for an additional cost on extra bandwidth in every time interval. Though tenant A's cost per time interval has increased, its application can enjoy a better performance. As shown in the bottom right of Fig. 1, it takes only 2 time intervals to finish the task by paying for extra best-effort bandwidth, and its total cost is also reduced because the price for best-effort bandwidth is cheaper than VM instance.

In fact, best-effort bandwidth shares should not only be charged for, they should be charged with *differentiated*, rather than uniform, prices. An intuitive rationale for differentiated pricing is that *a tenant may wish to accept a higher price to obtain a larger best-effort share*, so that better application performance can be attained. With the design and implementation of a bandwidth allocation algorithm, a larger best-effort share of the residual bandwidth on a network link can be allocated to a tenant, and the size of its allocated share depends on the price at which the tenant is willing to pay for.

Charging for Best-Effort Bandwidth Shares Simplifies the Design of Bandwidth Allocation Policies. Without explicit and differentiated pricing for best-effort bandwidth shares, it has been quite a challenge to allocate bandwidth *fairly* across competing tenants. This is due to a number of desirable properties that need to be maintained, such as strategyproofness, communication pattern independence, and symmetry [4]. By allowing an IaaS cloud provider to charge its tenants for both guaranteed and best-effort bandwidth shares explicitly, and by establishing differentiated pricing for best-effort bandwidth shares, we have not only provided the flexibility and power for a tenant to obtain a guaranteed or larger share of bandwidth, but also simplified the design of fair bandwidth allocation policies. We call the simpler design principle *cost dominant fairness*, which stipulates that bandwidth should be allocated based on the total cost that a tenant incurs for running its applications in the cloud, including the costs incurred for both VMs and the bandwidth shares. With the principle of cost dominant fairness, of which a more rigorous definition will be presented later, desirable properties can be more easily achieved.

4 The Solution Space for Pricing Bandwidth

Before a comprehensive investigation on bandwidth allocation following the guiding principle of cost dominant fairness, we first answer the fundamental question about the cost: how should reserved and best-effort bandwidth shares be priced, respectively? Without a proper design of pricing, apart from failing to realize the objective, the IaaS cloud provider may prevent a potential tenant from migrating to the cloud, due to the high cost incurred that is larger than the benefit gained by the tenant. In this section, we explore the entire solution space, and conclude with a pricing framework.

Pricing Guaranteed Bandwidth Shares. Reserved bandwidth are guaranteed to be available to a tenant. Since guaranteed-bandwidth traffic enjoys a higher priority than best-effort traffic, it should be priced higher than their best-effort counterpart. If p_r denotes the price for guaranteed-bandwidth traffic, and p_i for best-effort traffic, we have $p_r > p_i$.

But what is the unit of these prices, p_r and p_i? Since we are reserving and consuming bandwidth as a resource after all, the initial reaction is that these should be prices per unit of bandwidth, measured in, for example, Mbytes per second. Now consider two tenants, A and B. Both have reserved the same amount of bandwidth, but A reserved it for a period of time that is twice as long as B. Apparently, the bandwidth cost to A should be twice as much as the cost to B. This implies that the length of time when bandwidth is reserved should also play a role when the cost is to be computed, and p_r and p_i should be prices per unit of traffic volume, measured in, for example, GB. Such a unit of pricing conforms to the status quo in current cloud providers (e.g., Amazon EC2) when they charge their tenants for bandwidth usage going in and out of a datacenter, and has been adopted in [28] as well.

Returning to the issue of pricing guaranteed bandwidth shares, we emphasize that a tenant will need to pay for the *reserved* bandwidth that it requested over the entire reservation period, rather than the *actual* traffic volume it has transferred. This is due to the fact that once reserved, bandwidth

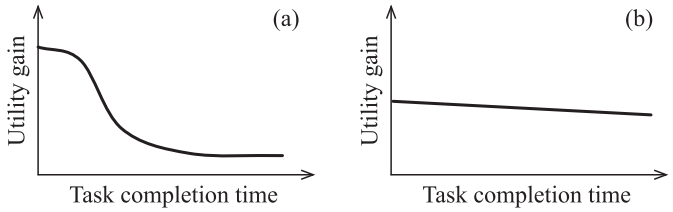


Fig. 2. Utility gains with different task completion times.

is set aside for the tenant regardless of usage, and will not be used by other tenants sharing the same network link.

Pricing Best-Effort Bandwidth Shares With Service Levels. How should differentiated prices for best-effort bandwidth shares be determined? An intuitive first-cut solution is to support multiple best-effort *service levels*, each with their own prices. More specifically, K service levels can be provided for the tenants to choose from, each associated with a price p_i per GB, with $p_i < p_{i+1}$, $i \in [0, K-1]$. Traffic transferred at more expensive levels will be allocated larger shares of bandwidth, proportional to their prices.

In such a differentiated pricing framework, the parameters of the number of provided service levels, K, and their prices, p_i, should be determined. Let us first consider the behaviour of tenants in our differentiated pricing framework. Let us assume that there exists a reservation value v_j for each tenant j. If tenant j's net reward — its utility gain (i.e., benefits) minus its total cost — falls below v_j after its cloud migration, it will not choose to migrate to the cloud. Then, the net reward that tenant j obtains by choosing the best-effort service level i can be represented as

$$N_{j,i} = U_{j,i} - p_i V_j - \mathrm{Cost}_j(\mathrm{VM}) \frac{V_j}{f_{j,i}}, \text{if } N_{j,i} \geq v_j, \tag{1}$$

which indicates excluding its cost on inter-VM traffic, which is $p_i V_j$, and its cost on reserving VM instances, i.e., $\mathrm{Cost}_j(\mathrm{VM}) \frac{V_j}{f_{j,i}}$, from the utility gain, represented by $U_{j,i}$, that tenant j can obtain by subscribing to service level i. Specifically, V_j is the volume of jth inter-VM traffic, $f_{j,i}$ is j's allocated bandwidth share, and $\mathrm{Cost}_j(\mathrm{VM})$ is the reservation cost per unit time. For example, if n_j identical VM instances with a per-VM price P are reserved to finish the task, $\mathrm{Cost}_j(\mathrm{VM}) = Pn_j$.

For a given tenant j, its utility gain $U_{j,i}$ is determined by its task completion time $\frac{V_j}{f_{j,i}}$, which is further determined by the service level i it subscribes to. A rational tenant will choose a service level that maximizes its expected net reward. More formally, for each tenant j, it will subscribe to level i such that

$$i = \arg\max N_{j,i}, \text{s.t. } N_{j,i} \geq v_j. \tag{2}$$

Ha *et al*. [32] has found that most cloud applications can be roughly categorized into two classes, based on whether an application is sensitive to its task completion time. Fig. 2 shows an illustration of the utility gains for both classes of applications. Since the total cost to a tenant includes its costs on both VMs and bandwidth, if a tenant's cost is dominated by its VMs and sensitive to its performance, it will have the incentive to reduce its task completion time by subscribing to a more costly service level, associated with a larger bandwidth share. On the other hand, if a tenant only has a few VMs and is not sensitive to its task completion time, it will prefer to subscribe to a service level that is less expensive, in order to minimize its costs on bandwidth. The majority of tenants who fall in between the two camps may very well choose a service level in the middle. Based on our back-of-the-envelope analysis, it is clear that multiple service levels — at least three — are necessary.

Having the tenants' strategies in mind, the cloud provider's problem is to maximize its revenue by choosing the optimal prices p_i for bandwidth shares at each best-effort service level.

Personalized Pricing for Best-Effort Bandwidth. Now the problem is how to determine the prices at each level. This is challenging as a cloud provider certainly wishes to maximize its revenue by charging the highest possible prices; while on the other hand, if the prices are too high, the net reward of each tenant may fall below the acceptable value, which then discourages the continued usage of the cloud service. To keep our pricing framework more practical, we propose to adopt a *personalized pricing* framework, where each tenant claims its own price $kp_b, k \in [1, \ldots, K]$, as a multiple of a low baseline price $p_b (p_b > 0)$ per GB of best-effort bandwidth. Since best-effort bandwidth may not assume a higher price than reserved bandwidth, we have $p_r > Kp_b$. The cost for the tenant consists of the the VM cost and bandwidth (both guaranteed and best-effort) cost, represented as

$$\mathrm{Cost}(\mathrm{VM}) * \frac{V}{f_k} + (p_r + kp_b) * V,$$

where V is the traffic volume and f_k is the actual bandwidth share it gets. We will have more discussion on the tenant cost in Section 7.

If we return to the concept of service levels, since the difference between p_b and p_r may potentially be large, a large number of service levels may exist in practice, given each tenant more flexibility and finer granularity. As a tenant chooses its own price, it has chosen its service level in effect. The cloud provider just needs to design and implement an allocation algorithm that allocates residual best-effort bandwidth, according to the tenant's total payment for both its VMs and its bandwidth, following the principle of cost dominant fairness.

As compared to the use of pre-defined service levels, our personalized pricing framework is simpler: the cloud provider no longer needs to determine individual prices for each service level; instead, a potentially larger number of levels are provided, and it is up to a tenant to choose a level (and its associated price) that it prefers. To improve its revenue, the cloud provider can still adjust both the baseline price p_b and the price for guaranteed bandwidth shares p_r over time.

5 Fair Sharing of Best-Effort Bandwidth

In this section, we proceed to discuss how to actually allocate the bandwidth resources to multiple tenants in achieving cost dominant fairness, by a comprehensive exploration of all the possibilities.

Fairness Criteria. While traffic from tenants with different levels in the best-effort service type need to share the residual bandwidth in the datacenter network, a proper network sharing policy is required to achieve fairness among all traffic flows. Before we discuss the sharing policy, we first need to precisely describe what, exactly, is the notion of fairness that we wish to achieve. With cost dominant fairness, the fairness criteria can be summarized as follows:

◇ When tenants are paying the same price per unit of traffic, their network-wide bandwidth shares should be proportional to their costs on reserving communicating VMs per unit time, i.e.,

$$f_{j_1,i} : f_{j_2,i} = \text{Cost}_{j_1}(\text{VM}) : \text{Cost}_{j_2}(\text{VM}), \quad \forall i \in [0, K-1].$$

◇ For tenants within different service levels, if they have incurred the same cost on reserving communicating VMs per unit time, their network-wide bandwidth shares should be dominated by (i.e., proportional to) their costs on inter-VM data transmission per unit of traffic, represented as

$$f_{j_1,i_1} : f_{j_2,i_2} = p_{i_1} : p_{i_2}, \quad \text{if} \quad \text{Cost}_{j_1}(\text{VM}) = \text{Cost}_{j_2}(\text{VM}).$$

With the definition of cost dominant fairness, we consider not only the service level a tenant subscribed to, but also the number of communicating VMs a tenant has payed for, as it is certainly unfair if two tenants get exactly the same network-wide bandwidth share when one has 100 VMs communicating with one another while the other one has only 2. A tenant should be given larger bandwidth resource if it subscribed to a more costly service level, or it has launched more VMs. The concept of cost dominant fairness lays the foundation of this paper. It tightly couples the notion of *fairness* with *costs* as cloud tenants share best-effort bandwidth in a datacenter network based on service levels. There is an incentive to ensure that the payment of each tenant corresponds to its received performance. It also gives a clear definition of fairness, such that the received performance by tenants is more predictable. The next critical challenge in this paper is to design proper sharing policies to achieve cost dominant fairness, based on a solid theoretical understanding of the intricacies in this objective.

Traditional sharing policies share bandwidth on each congested link equally, or in a weighted manner among flows, source-destination VM pairs, or among sources. Though they suffer from a variety of potential disadvantages [4], they all take the number of VMs as the proportionality parameter, and do not take the difference between traffic levels or types into account. To satisfy the requirements of cost dominant fairness, we need to redesign the sharing policies from scratch. In what follows, we explore the feasibility of assigning cost proportional weights at the tenant level, the VM-pair level and the flow level, respectively.

Per-Tenant Weight. Since we are trying to achieve fairness among tenants with different service levels at the network level, our first intuitive and natural solution is to take the tenant as the unit when bandwidth resources are allocated. That is, weighted fairness should be achieved among different tenants, each with its per-tenant weight.

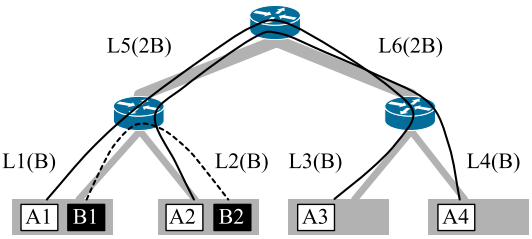


Fig. 3. An illustrative example of sharing policies, where lines connecting VMs show the traffic pattern of each cloud user.

As the aggregate bandwidth share of each tenant is not only affected by the number of flows, source-destination VMs, or sources, but also affected by the service level it subscribes to, the per-tenant weight should incorporate the service level prices. Guided by the fairness criteria, we are able to unify the tenant weights with different service levels and the different costs on VM reservations. Mathematically, if we assume that there are K levels in the best-effort service type, each associated with a price p_i per GB, $i \in [0, K-1]$, we can define a level weight to be $w_i = p_i$ for each service level i. We further define a VM weight w_v to be the price charged for a certain type of VM instance v per unit time, i.e., $w_v = p_v$. A tenant j's weight on VM reservation can be represented by $w_v^j = \sum_v w_v n_v^j$, where n_v^j is the number of type v VMs tenant j reserves during a charging period. Then for a tenant j who has subscribed to service level i, its per-tenant weight $w_{j,i}$ should be determined by

$$\frac{w_{j',i}}{w_{j,i}} = \frac{w_v^{j'}}{w_v^j}; \ and \ \frac{w_{j',i'}}{w_{j,i}} = \frac{w_{i'}}{w_i}, \ (if \ w_v^{j'} = w_v^j). \tag{3}$$

While this seems to be correct at first glance, a simple example can show that the allocation is unfortunately not cost dominant fair if we apply the per-tenant weight directly on congested links. For example, assume there are two tenants A and B who wish to finish the same task in the datacenter network, one initiated 4 VM instances while the other initiated 2, shown in Fig. 3. If both of them subscribe to the same service level, then A should get twice the bandwidth share in the network, dominated by their total costs per unit time. Now assume tenant A's weight is 2, then B's weight will be 1 based on Eqn. (3). Traffic flows from different source-destination VM pairs belonging to the same tenant will share the bandwidth in a weighted fashion.

On the congested link, $L1$ and $L2$ in this example, it is easy to see that the traffic between each source-destination VM pair of tenant A will get $B \times \frac{2}{3} \times \frac{1}{2} = \frac{B}{3}$ bandwidth, resulting in a total of $\frac{B}{3} \times 4 = \frac{4B}{3}$ bandwidth share for tenant A network-wide. The traffic generated by tenant B will get the remaining $B \times \frac{1}{3} = \frac{B}{3}$ bandwidth resources on both $L1$ and $L2$, which results in a ratio of $\frac{4}{1}$ regarding the two tenants' aggregated bandwidth shares. It can be seen that although it achieves congestion proportionality with ease, per-tenant weighted sharing fails to be fair at the network level.

Per-VM-Pair Weight. From this example, we can see that the cost on communicating VM reservation is obscurely reflected by the number of communicating source-destination VM pairs. We then wonder if we can assign weights at a finer granularity, say, on each communicating source-destination

VM pair, to achieve cost dominant fairness network-wide? In this case, weights assigned to each communicating VM pair belonging to each tenant will only be determined by the service level that tenant subscribes to, i.e., the weight for any communicating VM pair $X-Y$ belonging to tenant j who subscribed to service level i should be the price per unit of traffic at service level i, i.e.,

$$w_{j,i}^{X-Y} = p_i.$$

In the example, traffic between VM pairs $A1-A3$, $A2-A4$, and $B1-B2$ gets the same weight, as both tenants subscribe to the same service level, which results in a bandwidth share of $\frac{B}{2}$ for tenant A on both link $L1$ and $L2$. In total, tenant A gets B bandwidth share network-wide, which is twice the aggregate bandwidth share of tenant B, as required by our definition of cost dominant fairness.

Now, let us assume that tenant B has subscribed to a more expensive service level, in which he pays twice the price of tenant A per unit of traffic. Then both tenants should get the same network-wide bandwidth share based on cost dominant fairness. Using per VM-pair weight, traffic between VM pairs $A1-A3$ and $A2-A4$ should get the same weight that is half of the weight assigned to VM pair $B1-B2$, e.g., $w_{A,1}^{A1-A3} = w_{A,1}^{A2-A4} = 1$, and $w_{B,0}^{B1-B2} = 2$. Applying the per VM-pair weighted sharing, tenant A gets $\frac{B}{3}$ on both link $L1$ and $L2$, and tenant B gets the residual $\frac{2B}{3}$. In total, both tenants get the same network-wide bandwidth share as required by cost dominant fairness.

Per-Flow Weight. As it may be possible that there exist multiple flows between each VM source-destination pair, we wonder if we can provide weighted proportional shares on congested links in an even finer granularity, say, at the per-flow level. However, it turns out that a per-flow weight sharing policy does not provide cost dominant fairness. Suppose that there are two tenants subscribed to the same service level, and have the same cost on reserving communicating VMs in the datacenter network. Based on the requirement of cost dominant fairness, these two tenants should get exactly the same network-wide bandwidth share. However, with weighted share in the flow level, one tenant can easily grab a larger bandwidth share by initiating a larger number of flows between the two communicating VMs, without paying extra costs on traffic. Besides clearly violating the definition of fairness, the per-flow weighted sharing policy also violates the strategy-proofness property that is required in fair sharing, which we will discuss in detail soon.

With the three possibilities discussed above, we come to the conclusion that *per VM-pair weight assignment is our best choice towards achieving cost dominant fairness*. Before we get down to the details of bandwidth allocation in the next section, we analyze the desirable network sharing properties as follows.

Network Sharing Properties. As described in [4], there are a few desirable properties when sharing the datacenter network among multiple tenants. We find that with our pricing framework, assigning weights to communicating VM pairs achieves all the required properties.

Work Conservation. This property states that, if there is at least one cloud tenant who has traffic to send along a link, the link can not be idle. More precisely, a network sharing policy is said to be work-conserving if links in the datacenter network is either fully allocated, or it satisfies all demands. Since the bandwidth resource is shared proportionally in a weighted fashion, which always saturate the available residual bandwidth, the allocation is work conserving. Bandwidth that is unused, because the tenant needs less than its share or because part of its traffic is bottlenecked elsewhere, is re-apportioned among VM pairs belonging to other tenants of the link in proportion to their weights.

Symmetry. This property states that the allocated bandwidth should not depend on the direction of each flow. For example, suppose there is one VM sending traffic to two other VMs on a congested link, the allocated bandwidth between each VM pair should be the same if we change the role of source and destination VMs, i.e., letting the two receiving VMs sending traffic to the originally source VM. Without explicit pricing of best-effort bandwidth, these properties can only be achieved with complex weight calculations [4]. With the per VM-pair weighted sharing policy, this property is achieved as the bandwidth share is only determined by the number of communicating VMs and the service level a tenant subscribed to, but not the direction of its traffic.

Strategy-Proofness. What this property implies is that the network sharing policy should ensure that cloud tenants cannot improve their allocations by lying about their demands [33]. One can see that the per-flow network sharing policy fails to achieve the strategy-proofness property, as two VMs can increase the allocation between them by simply instantiating more flows. However, our per VM-pair weighted fairness satisfies this property, since a cloud tenant's utility remains the same for any demand declaration given the chosen service level and the number of communicating VMs. With the involvement of prices, rational tenants will not try to grab more bandwidth allocation by subscribing to more costly service levels, simply because more costly levels are associated with higher prices, and hence indicating more costs to the cloud tenant.

Network Proportionality. This property implies that the network resource should be shared among cloud tenants based on their payments, just as any other resource in the cloud. This property is the most critical property in a fair network sharing policy, and it also is the most challenging one to achieve. It is shown in [4] that even with very complicated weight assignment mechanisms, the best achievable proportionality is its relaxed variant, such as congestion proportionality. With cost dominant fairness, we are able to improve the proportionality towards the network level with the per VM-pair weighted sharing policy. It achieves better network proportionality than just the link level or congestion level, yet with a very simple design.

6 Allocating Bandwidth With Per VM-Pair Weights

In this section, we present a distributed rate allocation algorithm that achieves weighted proportional fairness across traffic from different VM pairs.

With the weight of traffic between each VM-pair considered, an allocation of rates $\vec{f} = \{f_{X-Y}, \forall X-Y\}$ is proportional fair if it is feasible, i.e., all assigned rates are

non-negative, and the sum of assigned rates on each link is smaller than the link capacity, and if for any other feasible allocation $\vec{f}'$, the aggregate of proportional changes is zero or negative

$$\sum_{\{X-Y\}} w^{X-Y} \frac{f'_{X-Y} - f_{X-Y}}{f_{X-Y}} \leq 0, \tag{4}$$

where w^{X-Y} is the weight for traffic from VM X to Y. It is proved in the classic network utility maximization framework that, there exists one unique proportional fair allocation for a given network, and such allocation is obtained by maximizing the sum of utilities of all VM pairs over the set of feasible allocations, with the utility function of each VM pair being the $\log$ function of the source rate [34].

We consider a datacenter network with L links, each with a residual capacity c_l to be allocated to all traffic in the best-effort service type. Let $\{X-Y\}$ represent the set for all VM pairs, in which traffic are transmitted at rate f_{X-Y} for each VM pair, using a fixed set of links $L(X-Y)$ in its path, based on the routing protocol used in the datacenter network. If we associate each source $X-Y$ a utility function $U_{X-Y}(f_{X-Y}) = \log(f_{X-Y})$, the weighted proportional fair bandwidth allocation for all flows can be obtained by solving the following optimization problem:

$$\max_{\vec{f} \geq 0} \sum_{\{X-Y\}} w^{X-Y} \log(f_{X-Y}) \tag{5}$$

$$\text{s.t.} \sum_{X-Y: l \in L(X-Y)} f_{X-Y} \leq c_l, \ \forall l, \tag{6}$$

where Eqn. (6) states the capacity constraints for all links.

Based on dual decomposition, we can define the Lagrangian as

$$L = \sum [w^{X-Y} \log(f_{X-Y}) - \lambda^{X-Y} f_{X-Y}] + \sum_l \lambda_l c_l, \tag{7}$$

where $\lambda_l \geq 0$ is the Lagrange multiplier (link price) associated with the linear rate constraint on link l, and $\lambda^{X-Y} = \sum_{l \in L(X-Y)} \lambda_l$ is the aggregate path congestion price of those links used by the traffic between $X-Y$. Since the first term in the Lagrangian function is separable in $X-Y$, the $X-Y$th Lagrangian to be maximized by each VM-pair $X-Y$ for the given $\vec{\lambda}^*$ is

$$\max_{f_{X-Y} \geq 0} w^{X-Y} \log(f_{X-Y}) - \lambda^{X-Y} f_{X-Y}, \tag{8}$$

with the unique optimum obtained by

$$f^*_{X-Y}(\lambda^{X-Y}) = \frac{w^{X-Y}}{\sum_{l \in L(X-Y)} \lambda_l} = \frac{w^{X-Y}}{\lambda^{X-Y}}. \tag{9}$$

The master dual problem is

$$\max_{\vec{\lambda}} \sum w^{X-Y} \log \left(\sum_{l \in L(X-Y)} \lambda^{X-Y} \right) - \sum_l \lambda_l c_l \tag{10}$$
$$\text{s.t. } \lambda_l \geq 0, \ \forall l.$$

The dual problem can be solved by using the gradient projection method, where the dual variables are adjusted in the opposite direction to the gradient

$$\lambda_l(t+1) = \left[\lambda_l(t) - \alpha \left(c_l - \sum_{X-Y: l \in L(X-Y)} f^*_{X-Y}(\lambda^{X-Y}(t)) \right) \right]^+, \tag{11}$$

where t is the iteration index, $\alpha \geq 0$ is a sufficiently small positive step-size, and $[\cdot]^+$ denotes the projection onto the nonnegative orthant [35].

With this solution, the source of each VM pair optimizes its own sending rate by solving problem (8), for the given aggregate path congestion price used by the certain traffic; and links in the network update their corresponding link prices using Eqn. (11) in each iteration, based on the weights of VM pairs going through each link. Since the sources may be located with different distances from the links in the datacenter network, a certain link price may be probed by different sources at different rates, and also such feedback will reach their destinations after variable delays in an unknown order. It is hard to ensure that the updates at both flow sources and links are synchronized in a datacenter network. Therefore, we propose to allocate the bandwidth among each VM pair fairly in an *asynchronous* and *distributed* manner, which is proved to be optimal and converging [36].

The key idea is that each source updates its sending rate at time intervals $T_{X-Y} \subseteq \{1, 2, \ldots, \}$, by solving the optimization problem (8) based on its current knowledge of the link prices. That is, at time $t \notin T_{X-Y}$, $f_{X-Y}(t+1) = f_{X-Y}(t)$. Similarly, each link updates its link price at time $T_l \subseteq \{1, 2, \ldots, \}$. At time $t \notin T_l$, the link prices remain the same. When a source or a link tries to update its sending rate or link price, it uses an *estimated aggregate path congestion price* $\hat{\lambda^{X-Y}}(t)$ or *estimated source rates* $\hat{f_{X-Y}}(t)$ by averaging the last received data, i.e.,

$$\begin{aligned} \hat{\lambda^{X-Y}}(t) &= \sum_{t'=t-t_0}^{t} x_{t'} \hat{\lambda^{X-Y}}(t'), \text{and} \\ \hat{f_{X-Y}}(t) &= \sum_{t'=t-t_0}^{t} x_{t'} \hat{f_{X-Y}}(t'), \end{aligned} \tag{12}$$

where $\sum_{t'=t-t_0}^{t} x_{t'} = 1$.

To summarize, our bandwidth allocation algorithm takes the weights of traffic originated from a VM instance to other VM instances and the feedback from all links used by its traffic as input, and decides the optimal bandwidth share for the originating traffic belonging to each VM pairs. Shown in Algorithm 1, traffic sources and links in the datacenter network update their sending rates and link prices, respectively at different times, based on their estimations by using the current knowledge, and broadcast their updated values to the network, until satisfying the termination criterion.

More specifically, to implement and enforce the bandwidth allocation, the server hosting a sender VM can implement a shim layer between the VM and the network interface, using software-based token bucket filters to limit the rate of each VM-pair, similar to Seawall [3]. The shim will

piggyback the information of calculated rate ($f_{X-Y}(t+1)$ at time $t \in T_s$) on the packets (from VM X to Y), by reusing unused bits in existing packet headers or simply adding extra header. When packets traverse along the path, the VM-pair rate information will be available to switches. Each switch retrieves the VM-pair rates from the packets passing by and updates each link price periodically (`line 10`), which are feasible given the popularity of Software-Defined Networking (SDN) in datacenters. The calculated link prices will be sent in separate packets to the source VMs, which will then be used at the shim layers to update rate limits (`line 4`).

Algorithm 1. The Bandwidth Allocation Algorithm

1: **Initialize** (Set $t = 0$ and $\lambda_l = \epsilon, \forall l$)
2: **while** $|\lambda(t+1) - \lambda(t)| > \delta$ **do**
3: Each VM stores the newly received t_0 link prices in its local memory, with the older ones being replaced.
4: Each link stores the newly received t_0 source rates that go through this link in its local memory, with the older ones being replaced.
5: **if** Time $t \in T_s$ **then**
6: Each source chooses a sending rate $f_{X-Y}(t+1) = \frac{w^{X-Y}}{\hat{\lambda^{X-Y}}(t)}$, where $\hat{\lambda^{X-Y}}(t)$ is obtained by Eqn. (12).
7: Each source of $X-Y$ broadcasts the current source rate to all the links in its path.
8: **end if**
9: **if** Time $t \in T_l$ **then**
10: Each link updates its link price $\lambda_l(t+1) = \left[\lambda_l(t) - \alpha(c_l - \sum_{X-Y:l \in L(X-Y)} \hat{f_{X-Y}}(t))\right]^+$, where $\hat{f_{X-Y}}(t)$ is obtained by Eqn. (12).
11: Each link l broadcasts its new link prices to all the sources that go through it.
12: **end if**
13: **end while**

7 Billing Towards Cost Dominant Fairness

It is not difficult to find out that even with per VM-pair weights, there is still no guarantee that users in the same service level will get exactly the same bandwidth share system-wide. It is even possible that users in cheaper levels can get more aggregated bandwidth share than users from more costly levels, as the VM placement and current network status will affect the actual aggregated bandwidth share of a user.

The underlying reason is that the weight proportionality guaranteed with per VM-pair weight assignment is limited to the group of VM pairs that share the same bottleneck link. Given a pair of VMs from tenant A and and a pair from tenant B, if their inter-VM traffic traverses through different bottleneck links (shared by different groups of flows), they are very likely to receive different bandwidth share even if they are in the same service level, because their bottleneck links have different degrees of congestion.

The problem is difficult, if not impossible, to solve from the cloud provider's perspective. In order to compute the optimal weight for each user, it is necessary to trace the locations of congested links precisely in the datacenter network with respect to traffic between each VM pair of every user, which is hard (and perhaps unnecessary) to achieve in practice. Even if the congested links are located, a sophisticated network sharing policy is required to achieve the fairness requirement without violating the work conservation property. FairCloud has also verified this argument, as all existing fairness bandwidth sharing policies in datacenter networks, including the ones proposed by the work itself, fail to achieve network-wide proportionality [4].

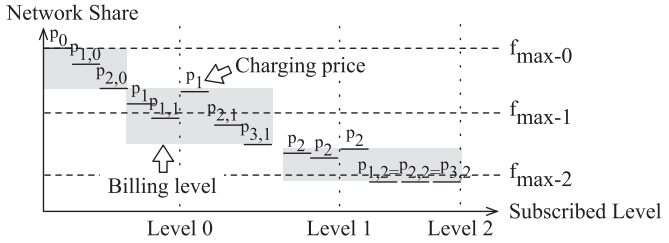


Fig. 4. An illustration of the proposed billing strategy.

We propose that, instead of solving the fairness problem directly, users subscribed to the same service level should be charged differently based on their actual achievable rate. The main idea in our billing strategy is that, users should be compensated accordingly if they paid more than what they are getting. In other words, the net reward a user obtains by completing a task at a service level should be unrelated to the underlying traffic pattern, which results in different bandwidth shares among users at the same service level.

More formally, let us get started from the most costly level, i.e., level 0. Assume the largest achievable network-wide bandwidth share across all users is f_{max-0}, and the user who gets the largest network share at this level is charged by p_0, the price per traffic unit associated service level 0. Then literally the largest network share in other levels should be proportional to the price compared to level 0, i.e., $f_{max-i} = \frac{p_i}{p_0} f_{max-0}, \forall i$. Fig. 4 presents an illustration of our proposed billing strategy, with three subscription levels as an example, to be elaborated in what follows.

In real datacenter networks, it might be possible that a user in a certain level i gets larger bandwidth share than the literal largest bandwidth share in this level, f_{max-i}. In our billing strategy, those users will be charged by p_i, only paying for f_{max-i} of their bandwidth share. The reason is that although those users receive more bandwidth resources than the literal value, they only signed for the level with price p_i, and there is no reason to let users pay for the additionally obtained idle resources assigned by the system to preserve work conservation. As shown in Fig. 4, in the second shaded block, a user subscribed to level 1 but received bandwidth share greater than f_{max-1}, as represented by the uppermost short horizontal line. Despite receiving more bandwidth, this user will only be charged by p_1, as annotated in the figure with an arrow.

It might also be the case that users subscribed to a more costly level receive lower network-wide bandwidth share due to congestion than the actual largest bandwidth share of its higher level next (i.e., the next level with a larger index value which is less expensive). To deal with this case, we propose that users in this case will be categorized into the next higher level. If the largest network share in one level is smaller than that of a higher level, all users in this service level will be charged as users in the higher level. The

rationale is that although users may have subscribed to a more costly service level, if they do not receive more bandwidth share than users in less costly levels, there is no reason to charge them more money. In other words, cost incurred to a user should correspond to its received bandwidth share, which follows our cost dominant fairness policy. Still with Fig. 4 as an example, in the second shaded block, a user that originally subscribed to level 0 receives a bandwidth share that is less than the largest actual network share of level 1, as represented by the leftmost horizontal line in this block. This user will be categorized into level 1 subscription. With the same analysis as in the previous case, this user will be charged by p_1.

Now with the re-categorized service levels, how should users be charged if their received bandwidth share is smaller than the literal largest network shares? Let us assume that there are n_i users who subscribed to each service level i, and the actual achievable network-wide bandwidth share for each user is $f_{j,i}, j \in [1, n_i]$. Without loss of generality, we can reorder the indexes of users such that $f_{j,i} \geq f_{j+1,i}, j \in [1, n_{i-1}]$. For user j at level i who gets the bandwidth share $f_{1,i+1} < f_{j,i} \leq f_{max-i}$, what our proposed billing strategy does is that the user will be charged based on a discounted price $p_{j,i}$, such that

$$c_j * \frac{V_j}{f_{j,i}} + \text{Cost}_j(\text{VM}) * \frac{V_j}{f_{j,i}} + p_{j,i} * V_j = c_j * \frac{V_j}{f_{1,i}} + \text{Cost}_j(\text{VM}) * \frac{V_j}{f_{1,i}} + p_i * V_j. \quad (13)$$

In Eqn. (13), c_j reflects the tolerance of a user j in the task completion time, i.e., the larger c_j is, the more urgent the task is. V_j states the total traffic incurred by completing user j's task, and $\text{Cost}_j(\text{VM})$ shows the costs incurred to user j by VM reservation. What this equation implies is that, user j should be charged based on a price such that his net reward remains the same if his task is completed by getting the largest possible network share at this service level. From Eqn. (13), we can obtain the actual billing price to user $j, j \neq 1$, at service level i as

$$p_{j,i} = p_i - [c_j + \text{Cost}_j(\text{VM})]\left[\frac{1}{f_{j,i}} - \frac{1}{f_{1,i}}\right]. \quad (14)$$

As illustrated in Fig. 4, in the first shaded block, two users in level 0 belong to the case that their received bandwidth share is smaller than the largest network share f_{max-0}. They will receive their discounted prices, $p_{1,0}$ and $p_{2,0}$, according to Eqn. (14). In the second shaded block, a user original subscribed to level 0 receives a bandwidth share smaller than f_{max-1}, represented by the horizontal line annotated with $p_{1,1}$. This user will first be recategorized to level 1 user, as discussed in the previous case, and then be charged with a discounted price $p_{1,1}$.

8 Experimental Results

To enforce per-VM-pair weights, similar to Seawall [3], our proposed bandwidth allocation algorithm can be implemented as rate controlled logical tunnels at the shim layer between the VMs and the network interface, in all the physical servers across the datacenter network. In order to evaluate our bandwidth allocation algorithm, we have implemented it in the Mininet 2.0 emulation testbed [39], and evaluated it over a $k = 4$ fat-tree topology with 4 pods, 16 end hosts, and 20 switches, with a link capacity of 1 Mbps on all the links in the network. To facilitate our evaluation, we use RipL, a Python library to build our OpenFlow controller that supports ECMP as the routing protocol, and records the path that each flow follows in the fat-tree topology. In addition, we use a network bandwidth monitoring tool called *Bandwidth Monitor NG* (`bwm-ng`) to measure the edge load at each network interface. The edge load statistics obtained by our bandwidth monitor are provided as input to our implementation of our bandwidth allocation algorithm every second.

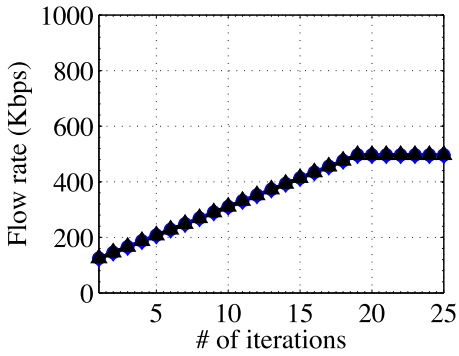


Fig. 5. The convergence of flow rates with a small group of 4 VM pairs, each with one flow.

To make it simpler to control the sending rates, we use UDP as the transport protocol for all our flows in the network, and use the `iperf` tool as the traffic generator, according to the flow rates computed by our algorithm. The main objective in our experiments is to evaluate the effectiveness of our bandwidth allocation algorithm to achieve weighted proportional fairness in an emulated fat-tree datacenter network. When comparisons are called for, we choose to compare with both Hedera [37] and MPTCP [38] (specifically the congestion control part), for the same set of source-destination VM pairs in our experiments. Hedera has been proposed to adaptively schedule active flows to non-conflicting paths to maximize the aggregated network utilization without using multiple paths, and MPTCP is the most representative multi-path transport protocol, implemented in the Linux kernel.

Bandwidth Allocation: Convergence. We first present whether or not flow rates will converge over time with our bandwidth allocation algorithm. We run our experiment with a small group of 4 flows between their respective VM pairs with equal weights, and Fig. 5 presents a visual illustration on how the rates of all four flows are able to converge after a period of 20 iterations with our algorithm, and the convergence leads to a proportional fair allocation of rates to all four flows.

Bandwidth Allocation: Weighted Proportional Fairness. To show the effectiveness of our bandwidth allocation algorithm on enforcing weighted proportional fairness, we run our experiments with a larger group of 10 flows with equal weights, and compare its results with both Hedera and MPTCP. Fig. 6 shows the respective rates of all 10 flows after convergence, as compared to both Hedera and

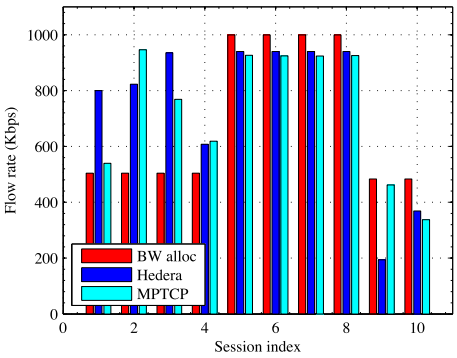


Fig. 6. The converged flow rates (of 10 flows) with our bandwidth allocation algorithm, as compared to both Hedera [37] and MPTCP [38].

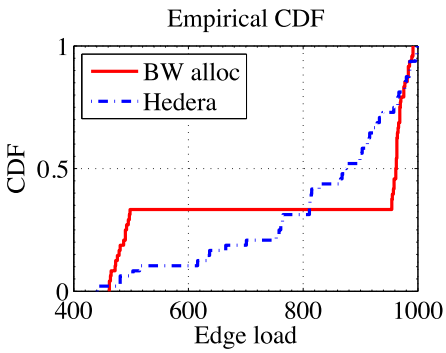


Fig. 7. The CDF of edge load statistics with 10 flows.

MPTCP. We can easily see that our algorithm has achieved weighted proportional fairness perfectly, with equal shares of bandwidth allocated to flows when they compete for a congested link. For flows with no competition, they are able to saturate the link capacities along their paths. In comparison, both Hedera and MPTCP are able to achieve similar flow rates as our algorithm when there is no competition among the flows; yet with competing flows, both fail to allocate the bottleneck bandwidth fairly according to the flow weights.

Combined with our results on the edge load CDF from all 48 edges in the network as shown in Fig. 7, we can see that our algorithm is able to utilize link bandwidth approximately as well as Hedera, yet with better fairness achieved among competing flows. Literally, our algorithm is able to deprive the resources unfairly allocated to the "rich" flows by both Hedera and MPTCP, and re-allocate them to the "poor" flows so that they are allocated equal bandwidth shares.

To magnify the illustration of our fairness comparison, Fig. 8 has singled out the only two pairs of sessions competing with each other for link capacities: Flow 1 versus 9; as well as Flow 4 versus 10. From this illustration, we can easily see the major difference between our proposed algorithm and Hedera/MPTCP with respect to fairness. The conclusion is crystal clear: our proposed algorithm is able to outperform both Hedera and MPTCP with respect to fairness between the competing flows.

Large-Scale Simulation. We further conducted simulations with a larger-scale to demonstrate the effectiveness and performance of our proposed bandwidth allocation algorithm towards cost dominant fairness. Particularly, we consider a datacenter of the fat-tree topology with 8 pods, which

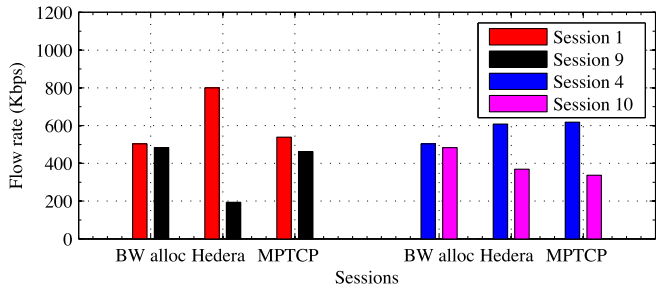


Fig. 8. In a three-way comparison with Hedera [37] and MPTCP [38], only the proposed bandwidth allocation algorithm has achieved weighted proportional fairness.

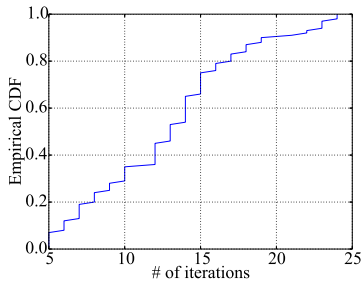


Fig. 9. The empirical CDF of the number of iterations for convergence with 100 flows.

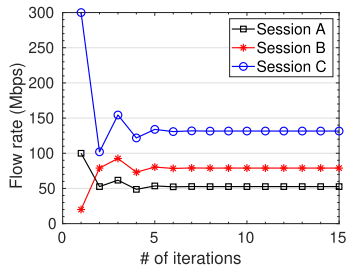


Fig. 10. Convergence of three sessions in simulation group 1.

interconnect 128 servers that can host more than 1,000 VMs. The link bandwidth is set as 1 Gbps at the edge and the network has an over-subscription factor of 1:1 with full bisection bandwidth. Our traffic load is generated as 100 inter-VM flows, with 30 percent as intra-pod and 60 percent across pods through the core switches. The starting rate of each flow is randomly set as a real value within the cap of link capacity. The weights of flows are also randomly specified in the range of (0,5], to differentiate their importance associated with the service levels that their users subscribe to.

Fig. 9 shows the convergence statistics of the 100 flows. As observed, the number of iterations required before convergence is 24 at most. 90 percent of the flows converge within 20 iterations, and 35 percent require no more than 10 iterations to converge. Fig. 10 shows the sending rate dynamics for three randomly selected flows given our bandwidth allocation mechanism. Starting with randomly generated flow rates, these flow sessions are able to converge to their respective bandwidth allocation that is regulated by cost dominant fairness. As observed in the figure, Session A, B and C eventually achieve the sending rates of 52.9, 79.4 and 131.6 Mbps, respectively. It is worth noting that Session A and B are allocated with proportional bandwidth to their weights of 2 and

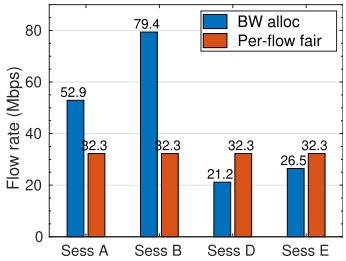


Fig. 11. Converged flow rates in comparison with per-flow fairness in simulation group 1.

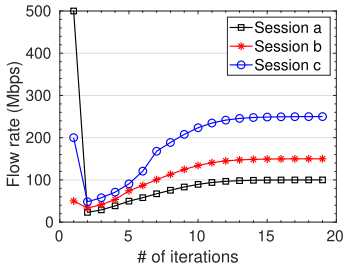


Fig. 12. Convergence of three sessions in experimental group 2.

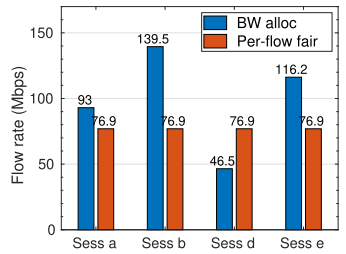


Fig. 13. Converged flow rates in comparison with per-flow fairness in simulation group 2.

3, but Session C with weight of 5 does not achieve this proportionality. This is because Session C traverses a different bottleneck link which is more crowded. We further investigate four flows that compete with each other for the same bottleneck link. The comparison of the allocated bandwidth among the four flows with our cost dominant fair allocation and per-flow fair allocation is illustrated in Fig. 11. Intuitively, with per-flow fairness in conventional TCP, all of the flows will achieve the same bandwidth, neglecting the factor of subscribed service level for best-effort bandwidth. In contrast, with our bandwidth allocation, the sessions with the weight proportionality of 2:3:0.8:1 achieve the same proportionality for their converged flow rates, satisfying the cost dominant fairness.

Similar analysis applies to Figs. 12 and 13 in our second simulation group. Fig. 12 demonstrates that the randomly selected four flows are able to converge fast with our bandwidth allocation mechanism, and their final allocated rates are proportional to their weights (2, 3, 5, respectively). Four flows sharing the same bottleneck link are singled out in Fig. 13 to magnify the illustration of fairness comparison. Given the weight proportionality of 2:3:1:2.5 for Session a, b, c and d, it is easily verified that their flow rates exhibit the same proportionality and cost dominant fairness has been achieved.

Finally, we have conducted another group of simulation to investigate flow rate convergence given different starting rates.

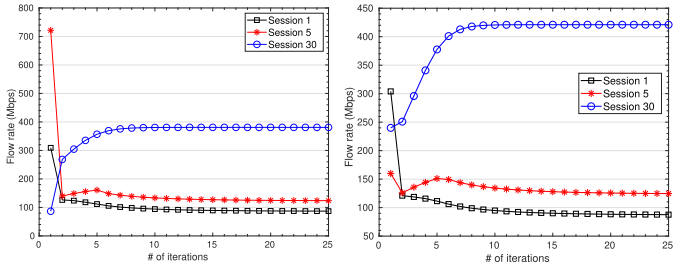


Fig. 14. Convergence of three sessions in simulation group 3, with different initial rates.

The two subfigures in Fig. 14 present the flow rate convergence of the same set of flows, under the same setting except for different starting rates. As observed, regardless of the initial rate, each of the flows will converge to the same sending rate, i.e., 81.7 Mbps for Session 1, 116.6 Mbps for Session 5 and 384.6 Mbps for Session 30, determined by its weight.

Discussion. While our work is focused more on theoretical model perspective and our current evaluation is for proof-of-concept, it would be interesting to see how the proposed framework works in the real datacenter environment of an IaaS provider. We present a personalized pricing framework, with the baseline price p_b for best-effort bandwidth and the price p_r for guaranteed bandwidth, allowing the cloud user to specify service level k. In this framework, how to set and adjust the unit prices p_b and p_r, as well as the VM reservation price, is a complementary direction to our current work. As bandwidth is now charged separately, it is intuitive that VM reservation price needs to be adjusted. For cloud users without inter-VM traffic, the VM reservation price should be intuitively lowered to some extent. For all the users with inter-VM traffic, it is reasonable to set the prices such that compared with the conventional pricing, they would be charged more only if they achieve better network performance (faster flow completion). From the perspective of the IaaS provider, to improve its revenue, the provider can make optimal decisions from time to time based on the behavior of the cloud users, and the game-theoretical framework would be a promising tool to investigate the interaction and dynamics between cloud users and the provider [40].

Another concern is the implementation of bandwidth allocation. We use a simple weight assignment scheme in our bandwidth allocation, followed by a discounted billing strategy, to realize our objective of cost dominant fairness. As an alternative, weight assignment and bandwidth allocation could be designed with more delicacy, which hopefully can eliminate the need of discounted billing for compensation. For example, we may use a more sophisticated weight assignment scheme based on the global knowledge of all the traffic in the network, and enforce the rate assignment by modifying the switches in the network core to control traffic with rate limiters and weighted fair queues. Our current design is driven by the ease of deployment, which only requires the hosts to limit rates based on feedback from the paths. Given that an IaaS cloud provider has full access to its datacenter network and SDN switches have gained popularity, we believe that a provider can flexibly switch to alternative strategies that are more intricate, when it wishes to mitigate the burden of discounted billing.

9 Concluding Remarks

With tenants leasing CPU and memory in the form of VMs but using inter-VM bandwidth in datacenter networks free of charge, irrespective of the definition of fairness for bandwidth sharing, it is very challenging to achieve such fairness with network-wide proportionality [4]. Existing work proposed to allocate bandwidth based on the number of VMs that tenants have paid for. In this paper, we strongly advocate that a tenant should be given the flexibility to obtain more bandwidth, by *pricing* inter-VM bandwidth explicitly and separately from pricing VM reservations, using the notion of *cost dominant fairness*. It stipulates that bandwidth should be allocated based on the total cost that a tenant incurs, and that a tenant will pay proportionally more if it enjoys better performance. The highlight of this paper is a study on how weighted proportional fairness can be satisfied using a distributed bandwidth allocator, based on a problem formulation using network utility maximization. Using the Mininet emulation testbed and simulations, we have shown that algorithm allocates bandwidth in a fair manner, and outperforms existing solutions with respect to fairness.

Acknowledgment

The research was supported in part by a grant from BoRSF-RCS under the contract LEQSF(2019-22)-RD-A-21, and in part by RGC GRF grants under the contracts 16206417 and 16207818.

References

[1] M. Zaharia, A. Konwinski, A. D. Joseph, R. Katz, and I. Stoica, "Improving MapReduce performance in heterogeneous environments," in *Proc. 8th USENIX Conf. Operating Syst. Des. Implementation*, 2008, pp. 29–42.

[2] A. Iosup, N. Yigitbasi, and D. Epema, "On the performance variability of production cloud services," in *Proc. 11th Int. Symp. Cluster Cloud Grid Comput.*, 2011, pp. 104–113.

[3] A. Shieh, S. Kandula, A. Greenberg, C. Kim, and B. Saha, "Sharing the data center network," in *Proc. 8th USENIX Conf. Netw. Syst. Des. Implementation*, 2011, pp. 309–322.

[4] L. Popa, G. Kumar, M. Chowdhury, A. Krishnamurthy, S. Ratnasamy, and I. Stoica, "FairCloud: Sharing the network in cloud computing," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 42, pp. 187–198, 2012.

[5] C. Guo *et al.*, "SecondNet: A data center network virtualization architecture with bandwidth guarantees," in *Proc. 6th ACM Int. Conf.*, 2010, pp. 1–12.

[6] H. Rodrigues, J. R. Santos, Y. Turner, P. Soares, and D. Guedes, "Gatekeeper: Supporting bandwidth guarantees for multi-tenant datacenter networks," in *Proc. 3rd Conf. I/O Virtualization*, 2011, pp. 1–6.

[7] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards predictable datacenter networks," in *Proc. ACM SIGCOMM Conf.*, 2011, pp. 242–253.

[8] D. Xie, N. Ding, Y. C. Hu, and R. Kompella, "The only constant is change: Incorporating time-varying network reservations in data centers," in *Proc. ACM SIGCOMM Conf. Appl. Technol. Archit. Protocols Comput. Commun.*, 2012, pp. 199–210.

[9] K. Jang, J. Sherry, H. Ballani, and T. Moncaster, "Silo: Predictable message latency in the cloud," in *Proc. ACM Conf. Special Interest Group Data Commun.*, 2015, pp. 435–448.

[10] T. Lam, S. Radhakrishnan, A. Vahdat, and G. Varghese, "NetShare: Virtualizing data center networks across services," Univ. California, San Deigo, CA, USA, Tech. Rep. CS2010-0957, 2010.

[11] J. Guo, F. Liu, D. Zeng, J. C. S. Lui, and H. Jin, "A cooperative game based allocation for sharing data center networks," in *Proc. IEEE INFOCOM*, 2013, pp. 2139–2147.

[12] J. Guo, F. Liu, H. Tang, Y. Lian, H. Jin, and J. C. S. Lui, "Falloc: Fair network bandwidth allocation in IaaS datacenters via a bargaining game approach," in *Proc. 21st IEEE Int. Conf. Netw. Protocols*, 2013, pp. 1–10.

[13] V. Jeyakumar, M. Alizadeh, D. Mazieres, B. Prabhakar, C. Kim, and A. Greenberg, "EyeQ: Practical network performance isolation at the edge," in *Proc. USENIX Conf. Netw. Syst. Des. Implementation*, 2013, pp. 297–312.

[14] H. Ballani, K. Jang, T. Karagiannis, C. Kim, D. Gunawardena, and G. O'Shea, "Chatty tenants and the cloud network sharing problem," in *Proc. USENIX Conf. Netw. Syst. Des. Implementation*, 2013, pp. 171–184.

[15] L. Popa, P. Yalagandula, S. Banerjee, and J. Mogul, "ElasticSwitch: Practical work-conserving bandwidth guarantees for cloud computing," in *Proc. ACM SIGCOMM Conf.*, 2013, pp. 351–362.

[16] J. Guo, F. Liu, J. C. S. Lui, and H. Jin, "Fair network bandwidth allocation in IaaS datacenters via a cooperative game approach," *IEEE/ACM Trans. Netw.*, vol. 24, no. 2, pp. 873–886, Apr. 2016.

[17] S. Liu, L. Chen, and B. Li, "Siphon: A high-performance substrate for inter-datacenter transfers in wide-area data analytics," in *Proc. Symp. Cloud Comput.*, 2017, pp. 646–646.

[18] F. Liu, J. Guo, X. Huang, and J. C. S. Lui, "eBA: Efficient bandwidth guarantee under traffic variability in datacenters," *IEEE/ACM Trans. Netw.*, vol. 25, no. 1, pp. 506–519, Feb. 2017.

[19] S. Liu, L. Chen, and B. Li, "Siphon: Expediting inter-datacenter coflows in wide-area data analytics," in *Proc. USENIX Annu. Tech. Conf.*, 2018, pp. 507–518.

[20] L. Chen, Y. Feng, B. Li, and B. Li, "Promenade: Proportionally fair multipath rate control in datacenter networks with random network coding," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 11, pp. 2536–2546, Nov. 2019.

[21] L. Chen, Y. Feng, B. Li, and B. Li, "Towards performance-centric fairness in datacenter networks," in *Proc. IEEE Conf. Comput. Commun.*, 2014, pp. 1599–1607.

[22] L. Chen, B. Li, and B. Li, "Barrier-aware max-min fair bandwidth sharing and path selection in datacenter networks," in *Proc. IEEE Int. Conf. Cloud Eng.*, 2016, pp. 151–160.

[23] L. Chen, B. Li, and B. Li, "Surviving failures with performance-centric bandwidth allocation in private datacenters," in *Proc. IEEE Int. Conf. Cloud Eng.*, 2016, pp. 52–61.

[24] L. Chen, Y. Feng, B. Li, and B. Li, "Efficient performance-centric bandwidth allocation with fairness tradeoff," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 8, pp. 1693–1706, Aug. 2018.

[25] L. Chen, W. Cui, B. Li, and B. Li, "Optimizing coflow completion times with utility max-min fairness," in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun.*, 2016, pp. 1–9.

[26] L. Chen, S. Liu, B. Li, and B. Li, "Scheduling jobs across geo-distributed datacenters with max-min fairness," in *Proc. IEEE Conf. Comput. Commun.*, 2017, pp. 1–9.

[27] L. Chen, S. Liu, B. Li, and B. Li, "Scheduling jobs across geo-distributed datacenters with max-min fairness," *IEEE Trans. Netw. Sci. Eng.*, vol. 6, no. 3, pp. 488–500, Third Quarter 2019.

[28] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "The price is right: Towards location-independent costs in datacenters," in *Proc. 10th ACM Workshop Hot Topics Netw.*, 2011, pp. 1–6.

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

[30] K. Nagaraj, D. Bharadia, H. Mao, S. Chinchali, M. Alizadeh, and S. Katti, "NUMFabric: Fast and flexible bandwidth allocation in datacenters," in *Proc. ACM SIGCOMM Conf.*, 2016, pp. 188–201.

[31] J. Guo, F. Liu, T. Wang, and J. C. Lui, "Pricing intra-datacenter networks with over-committed bandwidth guarantee," in *Proc. USENIX Conf. Usenix Annu. Tech. Conf.*, 2017, pp. 69–81.

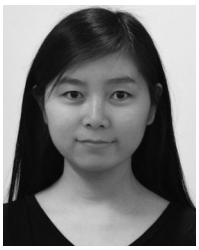
[32] S. Ha, S. Sen, J.-W. Carlee, Y. Im, and M. Chiang, "TUBE survey questions and demographics," Jan. 2012. [Online]. Available: http://scenic.princeton.edu/tube/tdpsurvey.html

[33] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica, "Dominant resource fairness: Fair allocation of multiple resource types," in *Proc. 8th USENIX Conf. Netw. Syst. Des. Implementation*, 2011, pp. 323–336.

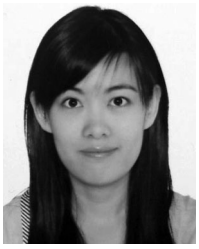
[34] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: Shadow prices, proportional fairness and stability," *J. Oper. Res. Soc.*, vol. 49, no. 3, pp. 237–252, 1998.

[35] D. P. Palomar and M. Chiang, "A tutorial on decomposition methods for network utility maximization," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1439–1451, Aug. 2006.

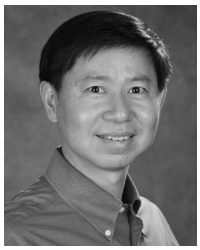
[36] S. H. Low and D. E. Lapsley, "Optimization flow control, I. Basic algorithm and convergence," *IEEE/ACM Trans. Netw.*, vol. 7, no. 6, pp. 861–874, Dec. 1999.
[37] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, and A. Vahdat, "Hedera: Dynamic flow scheduling for data center networks," in *Proc. 7th USENIX Conf. Netw. Syst. Des. Implementation*, 2010, Art. no. 19.
[38] C. Raiciu, S. Barre, C. Pluntke, A. Greenhalgh, D. Wischik, and M. Handley, "Improving datacenter performance and robustness with multipath TCP," in *Proc. ACM SIGCOMM Conf.*, 2011, pp. 266–277.
[39] N. Handigol, B. Heller, V. Jeyakumar, B. Lantz, and N. McKeown, "Reproducible network experiments using container-based emulation," in *Proc. 8th Int. Conf. Emerg. Netw. Exp. Technol.*, 2012, pp. 253–264.
[40] L. Zheng, C. Joe-Wong, C. W. Tan, M. Chiang, and X. Wang, "How to bid the cloud," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 71–84, 2015.



Li Chen received the BEng degree from the Department of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China, in 2012, and the MASc and PhD degrees from the Department of Electrical and Computer Engineering, University of Toronto, Toronto, Canada, in January 2015 and July 2018, respectively. She is an assistant professor with the Department of Computer Science, School of Computing and Informatics, University of Louisiana at Lafayette. Her research interests include big data analytics, machine learning systems, cloud computing, datacenter networking, resource allocation, and scheduling in networked systems.



Yuan Feng received the BEngr degree from the School of Telecommunications, Xidian University, Xi'an, China, in 2008, and the MASc and PhD degrees from the Department of Electrical and Computer Engineering, University of Toronto, Toronto, Canada, in 2010 and 2013, respectively. She is currently with Manulife Canada. Her research interests include optimization and design of large-scale distributed systems and cloud services.



Baochun Li (Fellow, IEEE) received the BEng. degree from the Department of Computer Science and Technology, Tsinghua University, Beijing, China, in 1995, and the MS and PhD degrees from the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, in 1997 and 2000, respectively. Since 2000, he has been with the Department of Electrical and Computer Engineering, University of Toronto, where he is currently a professor. He holds the Bell Canada endowed chair in computer engineering since August 2005. His research interests include cloud computing, distributed systems, datacenter networking, and wireless systems. He was the recipient of the IEEE Communications Society Leonard G. Abraham Award in the Field of Communications Systems in 2000. In 2009, he was a recipient of the Multimedia Communications Best Paper Award from the IEEE Communications Society, and a recipient of the University of Toronto McLean Award. He is a member of the ACM.

Bo Li (Fellow, IEEE) received the BEng (summa cum laude) degree in computer science from Tsinghua University, Beijing, China, and the PhD degree in electrical and computer engineering from the University of Massachusetts at Amherst, Amherst, Massachusetts. He is a chair professor with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. He held a Cheung Kong visiting chair professor with the Shanghai Jiao Tong University between 2010 and 2016, and was the chief technical advisor for ChinaCache Corp. (NASDAQ:CCIH), a leading CDN provider. He was an adjunct researcher with the Microsoft Research Asia (MSRA) (1999–2006) and with the Microsoft Advanced Technology Center (2007–2008). He made pioneering contributions in multimedia communications and the Internet video broadcast, in particular Coolstreaming system, which was credited as first large-scale Peer-to-Peer live video streaming system in the world. It attracted significant attention from both industry and academia and received the Test-of-Time Best Paper Award from IEEE INFOCOM (2015). He has been an editor or a guest editor for more than a two dozen of IEEE and ACM journals and magazines. He was the co-TPC chair for IEEE INFOCOM 2004.