Pricing Cloud Bandwidth Reservations under Demand Uncertainty

Di Niu, Chen Feng, Baochun Li
Department of Electrical and Computer Engineering
University of Toronto
{dniu, cfeng, bli}@eecg.toronto.edu

ABSTRACT

In a public cloud, bandwidth is traditionally priced in a pay-as-you-go model. Reflecting the recent trend of augmenting cloud computing with bandwidth guarantees, we consider a novel model of cloud bandwidth allocation and pricing when explicit bandwidth reservation is enabled. We argue that a tenant's utility depends not only on its bandwidth usage, but more importantly on the portion of its demand that is satisfied with a performance guarantee. Our objective is to determine the optimal policy for pricing cloud bandwidth reservations, in order to maximize social welfare, i.e., the sum of the expected profits that can be made by all tenants and the cloud provider, even with the presence of demand uncertainty. The problem turns out to be a large-scale network optimization problem with a coupled objective function. We propose two new distributed solutions — based on chaotic equation updates and cutting-plane methods — that prove to be more efficient than existing solutions based on consistency pricing and subgradient methods.

In addition, we address the practical challenge of forecasting demand statistics, required by our optimization problem as input. We propose a factor model for near-future demand prediction, and test it on a real-world video workload dataset. All included, we have designed a fully computerized trading environment for cloud bandwidth reservations, which operates effectively at a fine granularity of as small as ten minutes in our trace-driven simulations.

Categories and Subject Descriptors

K.6.2 [**Installation Management**]: Pricing and resource allocation; Performance and usage measurement; G.3 [**Probability and Statistics**]: Time series analysis

General Terms

Algorithms, Economics, Measurement, Performance

Keywords

Cloud Computing, Bandwidth Pricing, Distributed Optimization, Prediction, Time Series

SIGMETRICS'12, June 11–15, 2012, London, England, UK.

1. INTRODUCTION

Cloud computing delivers *Infrastructure as a Service* (IaaS) that integrates computation, storage and network resources in a virtualized environment. It represents a new business model where applications as *tenants* of the cloud can dynamically reserve *instances* on demand. However, a major risk to these tenants using cloud services is that unlike CPU and memory, bandwidth is not guaranteed in current-generation cloud platforms (e.g., Amazon EC2), leading to unpredictable network performance [6, 20]. A lack of bandwidth guarantee impedes cloud adoption by applications that require such guarantees, such as transaction processing web applications [14] and video-on-demand (VoD) applications [4]. The utility of tenants running these applications depends not only on the bandwidth usage, but more importantly on how many of their end-user requests are served with guaranteed performance.

With an ever-increasing demand for performance predictability, a recent trend in networking research is to augment cloud computing to explicitly account for network resources. In fact, datacenter engineering techniques have been developed to expand the tenant-cloud interface to allow bandwidth reservation for traffic flowing from a virtual machine (VM) in the cloud to the Internet [7, 12]. We envision that in future cloud platforms, bandwidth reservation will be a value-added feature that attracts tenants who seek bandwidth guarantees.

Unfortunately, even with cloud bandwidth reservation enabled, due to demand uncertainty, it is still difficult for a tenant to predict how much bandwidth it needs at a particular time. The usual approach of over-provisioning incurs high costs to tenants and does not really provide quantitative service guarantees. To promote guaranteed services, we believe that a new cloud service model should be introduced, in which a tenant simply needs to specify a percentage of its (bandwidth) demand to be served with guaranteed performance, which we call the *guaranteed portion*, while the rest of its demand will be served with best effort. It is then the cloud provider's responsibility to satisfy the guaranteed portion of the tenant with a high probability. Since the cloud provider has vast historical workload data, it can leverage statistical learning to predict tenant demands and make actual bandwidth reservations for the tenants.

In this paper, we study how to price the above guaranteed service. It is worth noting that usage-based pricing (pay-as-you-go) is not suitable for pricing bandwidth guarantees. For example, it is more costly to guarantee the performance of a tenant with bursty demand than a tenant with constant demand, even if they have incurred the same usage (number of bytes transferred). As a result, on top of the *usage fee*, the cloud should charge each tenant an extra *reservation fee*, depending on its unique demand statistics. Our objective is to fairly set such reservation fees, with the following

challenges. *First of all*, the cloud provider usually multiplexes tenant demands to save the service cost. Due to resource sharing, the absolute amount of bandwidth reserved for each tenant is unknown. It is a challenging question to find out each tenant's fair share in the aggregate service cost. *Second*, a pricing policy, when imposed to the market, may affect tenants demand; such demand change in turn affects pricing decisions, leading to potentially unstable iterations.

To overcome these difficulties, we define the reservation fee of each tenant as a function of its specified *guaranteed portion* instead of the absolute amount of bandwidth reserved. We also express each tenant's utility as a function of its *guaranteed portion*, which essentially measures the Quality of Service (QoS) at the tenant. Under this new model of pricing and utility, each tenant will choose a guaranteed portion to maximize its *surplus*, which is its utility minus price. Note that in reality, a tenant may choose a guaranteed portion *close to* 1 instead of being 1 out of cost concerns, while having the remaining demand served with best effort.

We study a cloud provider whose objective is to maximize the *social welfare* of the system, i.e., the total expected tenant utility under demand uncertainty minus the aggregate service cost. Although the cloud cannot know the exact form of utility at each tenant, it can affect each tenant's choice of guaranteed portion through pricing, and thus control the social welfare achieved. To handle the coupled cost function (due to multiplexing), we propose a novel algorithm based on *chaotic equation updates*, for which we provide a sufficient convergence condition. We further propose a distributed version of the *cutting-plane method* with guaranteed convergence. These methods are *step-size-free* and proved to be more efficient than traditional subgradient methods in simulations. In addition, we give explicit solutions to optimal pricing under certain special cases and point out the dependence of reservation pricing on demand statistics such as burstiness and covariances.

Since a main duty of the cloud provider is to reserve bandwidth for the tenants, demand forecast constitutes an important part in the reservation-based service. Toward this end, we propose a factor model to predict the expectations as well as covariances of tenant demands in the near future, based on principal component analysis (PCA). Finally, we evaluate the proposed algorithms on the workload traces of a real-world VoD system called *UUSee* [3]. We conduct trace-driven simulations of bandwidth reservation and algorithmic pricing based on demand prediction. The system is shown to operate effectively at a fine granularity (of as small as 10 minutes).

The remainder of the paper is organized as follows. We review related work in Sec. 2, and present our system model in Sec. 3. We formulate the problem of social welfare maximization in Sec. 4, where we outline the condition for optimal pricing and discuss its economic implications. To solve the optimal pricing problem distributively, in Sec. 5, we propose two algorithms: *chaotic price update* and the *distributed cutting-plane method*, and study their convergence performance. In Sec. 6, we present our statistical methods for demand forecast. We conduct trace-driven simulations in Sec. 7, and conclude the paper in Sec. 8.

2. BACKGROUND AND RELATED WORK

Cloud computing, e.g., Amazon EC2, is usually offered with usage-based pricing (pay-as-you-go) [6, 11]. Different from pay-as-you-go, resource reservation involves paying a negotiated cost to have the resource over a time period, whether or not the resource is used. Although suitable for delay-insensitive applications, pay-as-you-go is insufficient as a business model for bandwidth-intensive and quality-stringent applications like VoD, since no performance guarantees are provided in general.

The good news is that cloud bandwidth reservation is becoming technically feasible. There have been proposals on datacenter traffic engineering to offer elastic bandwidth guarantees for egress traffic from virtual machines (VMs) [12]. The idea of virtual networks has also been proposed to connect the VMs of the same tenant in a virtual network with bandwidth guarantees [7, 12]. Further, explicit rate control has been proposed to apportion bandwidth according to flow deadlines [22]. Such research progress has made the cloud more attractive to bandwidth-intensive applications such as video-on-demand and MapReduce computations that rely on the network to transfer large amounts of data at high rates [25]. Netflix, as a major VoD provider, moved its data store and video encoding and streaming servers to Amazon AWS [2] in 2010 [4].

To support guaranteed cloud services, we need new policies to price not only the bandwidth usage but also bandwidth reservations. Our pricing model is partially inspired by pricing electric power consumption and capacity reservation under demand uncertainty [19]. However, due to the computing capability and abundant workload data in the cloud, our bandwidth reservation pricing theory is essentially a distributed optimization problem based on statistical learning. Amazon Cluster Compute [1] allows tenants to reserve, at a high cost, a dedicated 10 Gbps network with no multiplexing. Instead of provisioning a fixed amount of capacity, we believe that tenants should be allowed to specify a guaranteed portion of demand, as a way to control QoS level, while cloud providers should dynamically vary bandwidth reservations based on demand predictions. Our approach has the unique advantage that tenants are exempted from demand estimation, for which they do not have expertise. In contrast, the cloud can easily access tenant demand history from online monitoring, and is computationally capable of accurate demand forecast.

Since pricing guaranteed portions critically depends on accurate estimates of demand statistics, we target applications with predictable demands, such as video access. As measurements show that video workload demonstrates regular diurnal periodicity [5, 17, 23, 24], various techniques have recently been proposed to forecast large-scale VoD traffic. Seasonal ARIMA models have been introduced in [16, 17] to predict non-stationary demand evolution at a fine granularity. Principal component analysis (PCA) has been proposed in [13] to extract video demand evolution patterns over longer periods (of weeks or months) and forecast coarse-grained daily populations. We combine the strengths of both approaches by finding the common factors driving the demand evolution of all tenants using PCA at a fine granularity. We then make predictions for individual tenants as regressions from factor forecasts obtained from seasonal ARIMA models. Unlike [13], our approach makes short-term predictions with a lead time of 10 minutes, enabling autoscaling of resource allocation.

Our optimal pricing algorithms are related to network utility maximization (NUM), which has been extensively studied in the past, with various distributed algorithms proposed. See [10, 18] for thorough surveys. Most of these algorithms assume no coupling in the objective function, and thus cannot be applied to our problem with a coupled cost term. One existing approach to handle coupled objectives is called *consistency pricing* [10, 21], which is based on dual decomposition and subgradient methods. However, subgradient methods suffer from the curse of step sizes, in that small steps incur big delays (many rounds of message exchanges between the cloud and tenants), while big steps yield big optimality gaps. Varying step sizes strategically is difficult in reality. In this paper, we propose two *step-size-free* algorithms: 1) chaotic price update, 2) the cutting-plane method. The first one is based on iterative equation updates instead of decomposition and achieves rapid conver-

gence under certain conditions. The second is a search algorithm with a guaranteed convergence speed.

3. A NEW TENANT-CLOUD AGREEMENT

Our system model is a generalization of the operation mode of the current cloud. Current cloud providers charge tenants a *usage fee* based on the number of bytes transferred in the past hour, and do not provide bandwidth guarantees. We extend this model to allow tenants to make reservations for bandwidth guarantees explicitly. The system operates on a short-term basis, e.g., based on hours or tens of minutes. At the beginning of each short period, each tenant specifies a *guaranteed portion* to guard against performance risks. The cloud decides the actual bandwidth reservation for tenants through demand estimation based on workload analysis, and charges both a usage and reservation fee. We now describe our system model in detail.

We consider one such short period, where N bandwidth-sensitive tenants are present. Suppose that in this period, tenant i's bandwidth demand is a random variable D_i (Mbps) with mean $\mu_i = \mathbf{E}[D_i]$ and variance $\sigma_i^2 = \mathbf{Var}[D_i]$. We assume the cloud can predict μ_i and σ_i based on demand history and share them to tenant i before the period starts. In our proposed market, the key commodity traded is a notion called the *guaranteed portion* instead of the absolute amount of bandwidth. Specifically, the tenants and cloud will comply to the following service agreement $\mathcal{S}_i(w_i, \epsilon, R_i)$:

- Before the period starts, each tenant i specifies a *guaranteed portion* $w_i \in [0, 1]$;
- The cloud guarantees w_i fraction of demand D_i with a high probability $1 - \epsilon$; outage is allowed to happen with a small probability ϵ, during which the bandwidth allocated to tenant i is limited to R_i.

The parameters ϵ and R_i are a part of a service level agreement (SLA) advertised by the cloud provider. We introduce the *risk factor* ϵ because for random demand, regardless of how much bandwidth is allocated, there exists a small risk of resource shortage.

Let q_i denote the actual bandwidth usage (realized data rate) in this period. Under a guaranteed portion w_i, service $\mathcal{S}_i(w_i, \epsilon, R_i)$ is supposed to lead to the following actual usage of tenant i:

$$q_i(w_i) = \begin{cases} w_i D_i, & \text{w.p. } 1-\epsilon, \\ \min\{w_i D_i, R_i\}, & \text{w.p. } \epsilon, \end{cases} \tag{1}$$

i.e., with probability $1 - \epsilon$ the actual usage q_i is a realization of w_i fraction of its demand D_i, while during outage (which happens with a small probability ϵ), the actual usage q_i is rationed by R_i.

Clearly, tenant i will choose w_i based on both the utility U_i and the price of guaranteeing w_i portion of its demand D_i. Unlike most prior work on network utility maximization [10] that assumes the utility U_i depends on a single variable such as rate, we model utility $U_i(q_i, D_i)$ of tenant i as a function of both actual usage q_i and the demand D_i. For example, a video content provider (or a VoD company) may have a linear utility gain (or revenue) $\alpha_i q_i$ from usage q_i and a convexly increasing utility loss $e^{A_i(D_i - q_i)}$ for the denied requests $D_i - q_i$, with α_i, A_i being tenant-specific parameters:

$$U_i\big(q_i(w_i), D_i\big) = \alpha_i q_i(w_i) - e^{A_i(D_i - q_i(w_i))}, \tag{2}$$

where the utility loss term can model the reputation degradation and potential revenue loss due to unfulfilled demand[1]. We assume U_i is *concave* and *monotonically increasing* in q_i.

The price for tenant i to use service $\mathcal{S}_i(w_i, \epsilon, R_i)$ is divided into two parts: a *usage fee* and a *reservation fee*. As most current cloud providers do, we assume uniform pricing for usage: each tenant pays $\$p$ for every unit bandwidth consumed. As a key departure from current clouds, we introduce a reservation fee, which is a function of the guaranteed portion instead of the absolute amount of bandwidth reservation: each tenant i is charged a price of $\$k_i w_i$ for having w_i portion of its demand guaranteed. We price the guaranteed portion w_i rather than the absolute bandwidth, because tenants usually have no idea about how much bandwidth they need. Instead, they can intuitively know how much percentage of guarantee is desired. This new business model frees each tenant from the computational burden of demand prediction: it simply submits its desired guaranteed portion w_i, while the cloud provider computes the actual bandwidth reservation as well as decides the reservation fee $k_i w_i$ for each tenant.

We define the *surplus* of tenant i as its utility minus its price:

$$S_i(w_i, p, k_i) := U_i\big(q_i(w_i), D_i\big) - p q_i - k_i w_i. \tag{3}$$

Given prices p and k_i, a rational tenant will choose a w_i to maximize its surplus S_i. Tenant i will not always choose $w_i = 1$ because when its demand is bursty, the price to guarantee 100% of D_i may be high. In this case, tenant i will choose a w_i close to 1 instead of being exactly 1, while the rest of its demand $(1 - w_i)D_i$ will be served with best efforts.

Based on tenant-specified guaranteed portions $w_1, \ldots, w_N$, the cloud should guarantee the demands $w_1 D_1, \ldots, w_N D_N$ for service. Denote $\mathbf{w} := [w_1, \ldots, w_N]^\mathsf{T}$. To realize the above service guarantees, the cloud provider needs to reserve a total *bandwidth capacity* of $K(\mathbf{w})$. Depending on the technology used, the value of K could vary significantly from one case to another. For example, a simple non-multiplexing technology is to reserve R_i capacity for each tenant i individually such that demand $w_i D_i$ is satisfied with high probability, i.e.,

$$\Pr(w_i D_i > R_i) < \epsilon, \tag{4}$$

and correspondingly, reserve capacity $K = \sum_i R_i$ in total. In contrast, a multiplexing technology will reserve capacity K for the tenants altogether such that the aggregate demand is satisfied with high probability, i.e.,

$$\Pr(\sum_i w_i D_i > K) < \epsilon, \tag{5}$$

and during outage (when $\sum_i w_i D_i > K$), the usage q_i of tenant i is rationed to R_i with $\sum_i R_i = K$. In both cases, K is an implicit function of $\mathbf{w}$ defined by the probabilistic constraints (4) and (5), respectively. To determine $K(\mathbf{w})$, the cloud provider must estimate the future demand statistics of all the tenants, and convert tenant-specified guaranteed portions $\mathbf{w}$ into the actual total bandwidth reservation K.

Similarly, the cloud provider has two kinds of service costs: usage and reservation costs. We assume tier-1 ISPs charge the cloud provider $\$b$ for every unit bandwidth actually used. Furthermore, reserving bandwidth capacity K will incur a reservation cost of $\$c(K)$. Due to multiplexing gain, to guarantee a similar service level, multiplexing will incur a lower K and thus a lower reservation cost $c(K)$ than without multiplexing.

[1]Even though the cloud provider may still be able to fulfill $D_i - q_i$ in a best-effort fashion, the tenant will have no knowledge if this is the case, and will not be able to factor it into its expected utility.

We define the *cloud profit* Π as the difference between its total revenue and total cost, i.e.,

$$\Pi(\mathbf{w}) := \sum_i \big(pq_i(w_i) + k_i w_i\big) - c\big(K(\mathbf{w})\big) - \sum_i bq_i(w_i). \quad (6)$$

4. PRICING TOWARDS SOCIAL WELFARE MAXIMIZATION

We study a cloud provider as a social planner whose objective is to maximize *social welfare* $W(\mathbf{w})$, which is defined as the total tenant utility minus the total service cost:

$$\begin{aligned} W(\mathbf{w}) &:= \sum_i U_i - c\big(K(\mathbf{w})\big) - \sum_i bq_i(w_i) \\ &= \Pi(\mathbf{w}) + \sum_i S_i(w_i, p, k_i) \end{aligned} \quad (7)$$

Under *random demands*, the cloud aims to decide a set of optimal guaranteed portions $\mathbf{w}^* = [w_1^*, \ldots, w_N^*]^\mathsf{T}$ for the tenants to maximize the *expected social welfare* by solving

$$\begin{aligned} &\max_{\mathbf{w}} E[W(\mathbf{w})] \\ &\text{s.t.} \quad \mathbf{0} \preceq \mathbf{w} \preceq 1. \end{aligned} \quad (8)$$

To solve (8), we first derive the expected social welfare in a simple *approximated* form. Note that $\mathbf{E}\big[U_i\big]$ is bounded as follows:

$$\begin{aligned} \mathbf{E}[U_i] &\geq (1-\epsilon)\mathbf{E}[U_i(w_i D_i, D_i)] + \epsilon \mathbf{E}[U_i(0, D_i)], \\ \mathbf{E}[U_i] &\leq (1-\epsilon)\mathbf{E}[U_i(w_i D_i, D_i)] + \epsilon \mathbf{E}[U_i(R_i, D_i)]. \end{aligned}$$

When the risk factor ϵ is small, we have

$$\mathbf{E}\big[U_i(q_i, D_i)\big] \approx (1-\epsilon)\mathbf{E}\big[U_i(w_i D_i, D_i)\big]. \quad (9)$$

To simplify notations, we define

$$\overline{U}_i(w_i) := \mathbf{E}\big[U_i(w_i D_i, D_i)\big], \quad (10)$$

which turns out to be *monotonically increasing* and *concave* in w_i under very mild technical conditions. Similarly, the expected usage of tenant i is

$$\mathbf{E}\big[q_i(w_i)\big] \approx (1-\epsilon)\mathbf{E}\big[w_i D_i\big] = (1-\epsilon) w_i \mu_i. \quad (11)$$

Therefore, the expected surplus of tenant i is

$$\begin{aligned} \mathbf{E}\big[S_i(w_i, p, k_i)\big] &= \mathbf{E}\big[U_i\big] - p\mathbf{E}\big[q_i\big] - k_i w_i \\ &= (1-\epsilon)\big(\overline{U}_i(w_i) - p w_i \mu_i\big) - k_i w_i, \end{aligned} \quad (12)$$

and the expected profit of the cloud provider is

$$\mathbf{E}[\Pi(\mathbf{w})] = (p-b)(1-\epsilon)\sum_i w_i \mu_i + \sum_i k_i w_i - c\big(K(\mathbf{w})\big). \quad (13)$$

Substituting the above into (7) gives the *expected social welfare* as

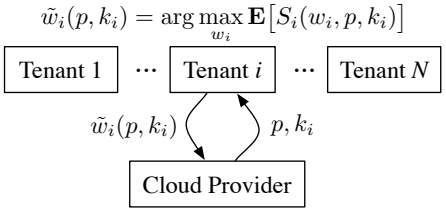
$$\mathbf{E}[W(\mathbf{w})] = (1-\epsilon)\sum_i \big(\overline{U}_i(w_i) - b w_i \mu_i\big) - c\big(K(\mathbf{w})\big). \quad (14)$$

4.1 An Equivalent Pricing Problem

In reality, although the cloud provider has full knowledge about its service cost $c(K(\mathbf{w}))$, it does not know the utility function of each tenant. In other words, maximizing $E[W(\mathbf{w})]$ in terms of $\mathbf{w}$ requires the cloud to know the utility U_i of each tenant and is infeasible. We now convert problem (8) into an equivalent pricing problem.

Note that the expected social welfare is also the sum of the expected cloud profit and the total expected tenant surplus, i.e.,

$$E[W(\mathbf{w})] = \mathbf{E}[\Pi(\mathbf{w})] + \sum_i \mathbf{E}[S_i(w_i, p, k_i)]. \quad (15)$$



Update p, k_i to increase social welfare

Figure 1: Iterative updates of prices and guaranteed portions.

Furthermore, when charged with prices p, k_i and facing a random demand D_i, a rational tenant will choose a guaranteed portion $\tilde{w}_i$ to maximize its expected surplus, i.e.,

$$\tilde{w}_i = \arg\max_{w_i} \mathbf{E}\big[S_i(w_i, p, k_i)\big], \quad (16)$$

which defines an implicit function $\tilde{w}_i(p, k_i)$ of the prices. The cloud can affect guaranteed portion choices $\tilde{\mathbf{w}} = [\tilde{w}_1, \ldots, \tilde{w}_N]^\mathsf{T}$ via appropriate choices of prices p, $k_1, \ldots, k_N$, and control the corresponding expected social welfare $\mathbf{E}[W(\tilde{\mathbf{w}})]$.

Therefore, the social welfare maximization problem (8) is converted into an equivalent **optimal pricing problem**:

$$\max_{p, k_1, \ldots, k_N} E[W(\tilde{\mathbf{w}})] = \mathbf{E}[\Pi(\tilde{\mathbf{w}})] + \sum_i \mathbf{E}[S_i(\tilde{w}_i, p, k_i)], \quad (17)$$

which, by combining (14) and (17), can be rewritten as

$$\max_{p, k_1, \ldots, k_N} (1-\epsilon)\sum_i \big(\overline{U}_i(\tilde{w}_i) - b\tilde{w}_i \mu_i\big) - c\big(K(\tilde{\mathbf{w}})\big), \quad (18)$$

where $\tilde{w}_i = \tilde{w}_i(p, k_i)$ is determined distributively by each tenant i via surplus maximization (16). Such a distributed optimization is illustrated in Fig. 1. Now the cloud provider does not need to know U_i: it simply charges tenant i the usage price p and reservation price k_i, and expect a $\tilde{w}(p, k_i)$ chosen by tenant i.

We denote the **optimal prices** that solve problem (18) as $p^*, k_1^*, \ldots, k_N^*$. The optimal pricing problem (18) is equivalent to the original problem (8), because by adjusting p and k_i, $\tilde{w}_i(p, k_i)$ can take any value in $[0, 1]$. In other words, the guaranteed portion $\tilde{w}_i(p^*, k_i^*)$ chosen by tenant i under optimal pricing p^*, k_i^* is exactly the guaranteed portion w_i^* that maximizes the expected social welfare, i.e., we have

$$w_i^* = \tilde{w}_i(p^*, k_i^*). \quad (19)$$

Therefore, once a set of optimal prices is obtained, we essentially have found a *decentralized solution* to expected social welfare maximization (8), which was originally impossible to solve.

Nonetheless, the optimal pricing problem (18) is not easy to solve either. At a first glance, (18) can be understood as a network utility maximization (NUM) problem [10] that may be solved via decomposition among the tenants. A closer look at (18) suggests that the term $c(K(\tilde{\mathbf{w}}))$ in the objective function may be coupled among all $\tilde{w}_i$'s, so that (18) cannot be decomposed into a set of subproblems, each solved at a tenant distributively. Coupling happens in the cost term when the cloud multiplexes tenant demands and books a capacity $K(\mathbf{w})$ for the aggregated demand. As shown in Fig. 1, the key to the solution is that the cloud provider must be able to update p and k_i towards the direction that increases $\mathbf{E}[W(\tilde{\mathbf{w}})]$. And a good price update algorithm should require fewer rounds of message-passing between the cloud and tenants before reaching optimality.

Before presenting the distributed solutions to (18) in Sec. 5, let us first provide a number of insights on how to make pricing poli-

cies, by checking the KKT conditions [8] that the optimal prices $p^*, k_1^*, \ldots, k_N^*$ must satisfy.

Proposition 1. *The optimal prices p^*, $k_1^*, \ldots, k_N^*$ must satisfy*

$$(1-\epsilon)(p^*-b)\mu_i + k_i^* - c'\big(K(\tilde{\mathbf{w}})\big)\cdot\frac{\partial K}{\partial w_i}\bigg|_{\mathbf{w}=\tilde{\mathbf{w}}} = 0, \quad \forall i, \tag{20}$$

where $\tilde{\mathbf{w}} = [\tilde{w}_1, \ldots, \tilde{w}_N]^\mathsf{T}$ with $\tilde{w}_i = \tilde{w}_i(p^*, k_i^*)$ given by (16).

Proof: Please refer to our technical report [15] for the proof. □

An inspection of (20) reveals that one set of optimal prices is

$$\begin{cases} p^* = b, \\ k_i^* = \partial c\big(K(\mathbf{w})\big)/\partial w_i\big|_{\mathbf{w}=\tilde{\mathbf{w}}}, \quad \forall i. \end{cases} \tag{21}$$

Although (21) is not the only set of optimal prices, our finding complies with the economic intuition that a welfare-maximizing cloud provider should charge marginal cost for both traffic usage and guaranteed reservation. In (21), we can also observe that k_i^* depends on $\tilde{\mathbf{w}}$, which in turn depends on $p, k_1^*, \ldots, k_N^*$. Due to such coupling, (21) is not yet a closed-form solution for the reservation price k_i^*.

4.2 No Multiplexing vs. Multiplexing All

To draw insights, we take a look at two special service technologies that may be adopted by the cloud: non-multiplexing and multiplexing across *all* the tenants. For simplicity, we assume a linear reservation cost (which will be relaxed later): $c(K) = \beta K$.

When multiplexing is not used, we can derive the optimal prices in a closed form from (21). Without multiplexing, recall that the capacity R_i is reserved for each tenant i individually, such that $\Pr(w_i D_i > R_i) < \epsilon$ and $K = \sum_i R_i$. When D_i is a Gaussian random variable (this assumption will be verified in Sec. 6), it is easy to check that

$$R_i(w_i) = \big(\mu_i + \theta(\epsilon)\sigma_i\big)w_i, \tag{22}$$

where $\theta(\epsilon) = F^{-1}(1-\epsilon)$ is a constant, with $F(\cdot)$ being the CDF of normal distribution $\mathcal{N}(0,1)$. Since the cost function is naturally decoupled among tenants, according to (21), the optimal prices are immediately given in a closed form by $p^* = b$ and

$$k_i^* = \beta\big(\mu_i + \theta(\epsilon)\sigma_i\big), \quad \forall i. \tag{23}$$

When multiplexing is used, however, optimal prices have no explicit solutions. Recall that with multiplexing, a capacity K is reserved to accommodate all the tenants together, such that the aggregate (instead of individual) demand is satisfied with high probability: $\Pr(\sum_i w_i D_i > K) < \epsilon$.

Since the random demands $D_1, \ldots, D_N$ of different tenants may be correlated, we denote ρ_{ij} the correlation coefficient of D_i and D_j, with $\rho_{ii} \equiv 1$. For convenience, let $\boldsymbol{\mu} = [\mu_1, \ldots, \mu_N]^\mathsf{T}$ and $\boldsymbol{\Sigma} = [\sigma_{ij}]$ be the $N \times N$ symmetric demand *covariance matrix*, with $\sigma_{ii} = \sigma_i^2$ and $\sigma_{ij} = \rho_{ij}\sigma_i\sigma_j$ for $i \neq j$.

Under Gaussian demands, K can be written as

$$\begin{aligned} K(\mathbf{w}) &= \mathbf{E}\big[\textstyle\sum_i w_i D_i\big] + \theta(\epsilon)\sqrt{\mathbf{Var}\big[\sum_i w_i D_i\big]} \\ &= \boldsymbol{\mu}^\mathsf{T}\mathbf{w} + \theta(\epsilon)\sqrt{\mathbf{w}^\mathsf{T}\boldsymbol{\Sigma}\mathbf{w}}. \end{aligned} \tag{24}$$

Substituting the above $K(\mathbf{w})$ into (21) gives $p^* = b$ and

$$k_i^* = \beta\bigg(\mu_i + \theta(\epsilon)\cdot\frac{\partial\sqrt{\mathbf{w}^\mathsf{T}\boldsymbol{\Sigma}\mathbf{w}}}{\partial w_i}\bigg|_{\mathbf{w}=\mathbf{w}^*}\bigg), \quad \forall i, \tag{25}$$

where $w_i^* = \tilde{w}_i(p^*, k_i^*)$. Clearly, with multiplexing, k_i^* is not given in a closed form yet, due to the coupled cost function.

We note that whether with or without multiplexing, $K(\mathbf{w})$ is a convex function and so is $c(K(\mathbf{w}))$. In fact, we can relax the linear cost assumption $c(K) = \beta K$, as long as $c\big(K(\mathbf{w})\big)$ is *strictly convex and monotonically increasing* for each $w_i \in [0,1]$.

There is an interesting connection between the non-multiplexing and multiplexing cases: the optimal solution for non-multiplexing can be used to bound the optimal solution for the multiplexing case. Specifically, the optimal prices $\{k_i^*\}$ of non-multiplexing upper-bound $\{k_i^*\}$ of the multiplexing case, whereas the optimal portions $\{w_i^*\}$ of non-multiplexing lower-bound $\{w_i^*\}$ of the multiplexing case. This is intuitive because multiplexing leads to a reduced cost $c(K)$, stimulating tenants to increase their choices of the guaranteed portion. The proof of this connection involves the use of Cauchy-Schwarz inequality and is omitted due to space limits. We will use this connection in our distributed algorithms.

4.3 Economic Implications

Condition (20) has several economic implications, which apply to a general cost function, although we may use the non-multiplexing case for explanation due to its simplicity.

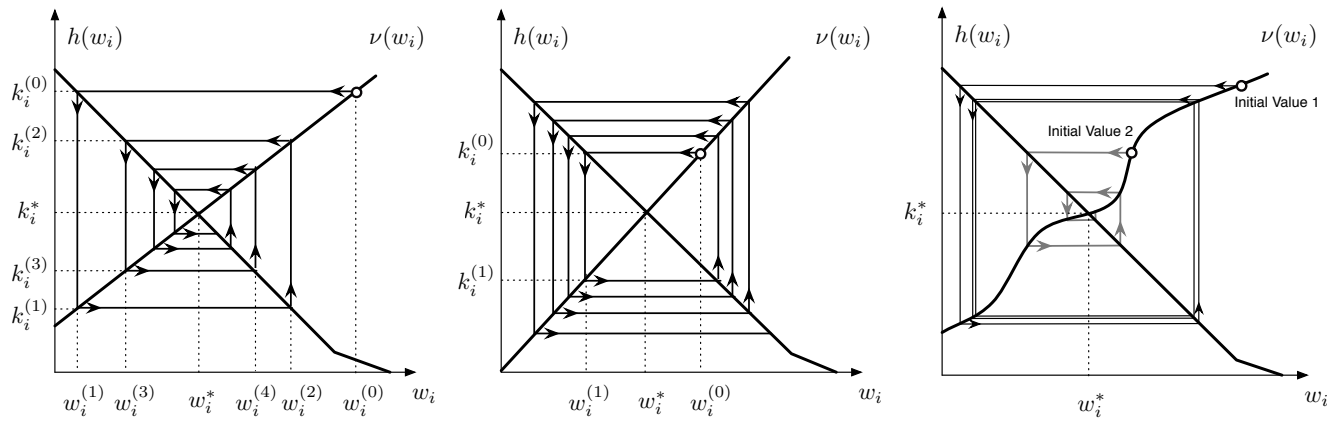
First, merely adopting a usage price p cannot maximize social welfare: when $k_i = 0$ for all i, there is no p that can satisfy (20). In other words, a positive reservation fee $k_i > 0$ is necessary to achieve welfare optimality, since in the presence of demand uncertainty, only k_i can incorporate a risk factor (e.g., σ_i in (23)) into pricing. This reveals that current cloud bandwidth pricing schemes are inefficient in terms of providing service guarantee against demand fluctuation. On the other hand, a usage price p is not necessary: even if $p = 0$, the expected welfare is maximized as long as k_i satisfies (20). In this case, the reservation fee can be raised to compensate the loss from no usage fee.

Furthermore, *heterogeneous* reservation prices $k_1, \ldots, k_N$ are necessary to achieve optimality, each k_i depending on the statistical characteristics of tenant i's demand D_i. This conforms to the intuition that tenants have different degrees of demand volatility, incurring different costs for service guarantees. For example, without multiplexing, k_i depends on σ_i in (23): the more bursty a tenant's demand D_i, the more capacity that must be reserved to guard against fluctuation, and thus the higher the price. In contrast, in terms of usage pricing, it is efficient enough to charge a homogeneous price p for every unit bandwidth consumed.

5. DISTRIBUTED SOLUTIONS

As has been noted, the main challenge to solving the optimal pricing problem (18) is that the reservation cost $c\big(K(\mathbf{w})\big)$ is coupled among all the tenants and is not decomposable in general. One existing approach to handle coupled objective functions is to find the dual problem of (18) and to decompose the dual among all the tenants and the cloud provider by introducing auxiliary variables. Such an approach is called *consistency pricing* [10, 21]. Subgradient methods are among the most popular techniques to update the prices towards the optimality of dual problems. However, they suffer from the curse of step sizes. For the final output to be close to the optimality, subgradient methods choose small step sizes to update k_i, leading to slow convergence and many iterations of message-passing between the cloud and tenants.

In this paper, we propose two novel *step-size-free* algorithms for price updates that can quickly converge to the optimality of (18). The first algorithm, called *Chaotic Price Update*, does not rely on decomposition at all: instead, it resorts to iterative equation updates based on the KKT conditions (20). The second algorithm, called the *Cutting-Plane Method*, relies on dual decomposition but does not update k_i using step sizes: it is essentially a search algorithm



(a) Convergence to Global Optimality (b) Divergence (c) 1: Periodic; 2: Converging.

Figure 2: Behavior of price update based on chaotic equations in a one dimensional case. "o" represents the starting point $(w_i^{(0)}, k_i^{(0)})$.

that locates $\{k_i^*\}$ until it is confined in a small region. We give a sufficient condition under which chaotic price update can achieve rapid convergence. The cutting-plane method, which is guaranteed to converge, is used to compensate chaotic price update when the latter is not converging. Note that our algorithms apply to a general convex cost function $c(K(\mathbf{w}))$ (in terms of $\mathbf{w}$) under any service technology (e.g., multiplexing tenant demands in groups).

5.1 Chaotic Price Update

Chaotic price update is based on alternated phases of price updates via the cost-price relationship (20) and the tenant surplus maximization equation (16).

Chaotic Price Update. Denote $\mathbf{w}^{(t)} := [w_1^{(t)}, \ldots, w_N^{(t)}]^\mathsf{T}$. Set $p \equiv b$. Set $k_i^{(0)} := \beta(\mu_i + \theta(\epsilon)\sigma_i)$ for all i and $\mathbf{w}^{(0)} = \mathbf{1}$. For $t = 0, 1, \ldots$, repeat

(1) **Distributed Surplus Maximization.** Pass p and $k_i^{(t)}$ to each tenant i, which returns:

$$w_i^{(t+1)} := \arg \max_{0 \le w_i \le 1} \mathbf{E}\big[S_i(w_i, p, k_i^{(t)})\big], \quad \forall i. \tag{26}$$

(2) **Price Update.** Set

$$k_i^{(t+1)} := c'\big(K(\mathbf{w}^{(t+1)})\big) \cdot \frac{\partial K(\mathbf{w})}{\partial w_i}\bigg|_{\mathbf{w}=\mathbf{w}^{(t+1)}}, \quad \forall i. \tag{27}$$

(3) If $\|\mathbf{w}^{(t+1)} - \mathbf{w}^{(t)}\| \le \xi$, return $\mathbf{w}^* = \mathbf{w}^{(t+1)}$, $k_i^* = k_i^{(t+1)}$.

The above algorithm starts by setting $k_i^{(0)}$ to be the k_i^* in the cost function without multiplexing. It then updates k_i and w_i alternately by setting the current prices $k_i^{(t)}$ to be the marginal reservation cost with the current $\mathbf{w}^{(t)}$, and by collecting the next $\mathbf{w}^{(t+1)}$ from tenants who maximize their surpluses given the current prices. Applying the KKT conditions to (16), step (1) can also be viewed as solving an equation

$$(1-\epsilon)\overline{U}_i'(w_i^{(t+1)}) = p(1-\epsilon)\mu_i + k_i^{(t)}, \tag{28}$$

for $w_i^{(t+1)}$. Since each tenant maximizes its expected surplus locally, the cloud provider does not have to know the utility function of each tenant, leading to an iterative distributed solution.

Compared with Lagrangian dual decomposition based on consistency pricing [10,21], chaotic price update represents a new way of handling coupled objectives. Since price updates are based on equation (27) rather than on updating Lagrangian multipliers, the algorithm is not concerned with the choice of step sizes that are required by subgradient methods.

We observe that the sequence $\{k_i^{(t)}\}$ produced by equations (27) and (28) could demonstrate significantly different behavior under different initial values $w_i^{(0)}$ and different forms of utility and cost functions. In other words, our algorithm demonstrates *chaotic* behavior, whose eventual outcome is sensitive to initial conditions and the structure of updating equations.

Our objective going forward is to analyze the behavior of $\{k_i^{(t)}\}$ and $\{\mathbf{w}^{(t)}\}$ and find out the conditions under which the algorithm can achieve fast convergence. To simplify notations, we define

$$h_i(w_i) \equiv (1-\epsilon)\big(\overline{U}_i'(w_i) - b\mu_i\big), \tag{29}$$

$$\nu_i(\mathbf{w}) \equiv \frac{\partial c\big(K(\mathbf{w})\big)}{\partial w_i}. \tag{30}$$

Recall that $\mathbf{w}^* := [w_1^*, \ldots, w_N^*]^\mathsf{T}$, where $w_i^* = \tilde{w}_i(p^*, k_i^*)$ is the guaranteed portion chosen by tenant i under optimal pricing. By the definition of $\mathbf{w}^*$, the optimality condition is

$$h_i(w_i^*) = k_i^* = \nu_i(\mathbf{w}^*), \quad \forall i. \tag{31}$$

Since we have set $p \equiv b$, (28) can be written as

$$h_i(w_i^{(t+1)}) := (1-\epsilon)\big(\overline{U}_i'(w_i^{(t+1)}) - b\mu_i\big) = k_i^{(t)}. \tag{32}$$

Thus, the **updating rules** (27) and (28) in chaotic price update can be rewritten as

$$h_i(w_i^{(t+1)}) = k_i^{(t)} = \nu_i(\mathbf{w}^{(t)}), \quad \forall i. \tag{33}$$

Let us illustrate the algorithm behavior using the special case of a *single* tenant. There are three scenarios where the algorithm can produce dramatically different results, as shown in Fig. 2. Since utility $u_i(w_i)$ is strictly concave and monotonically increasing in $[0, 1]$, and cost $c\big(K(\mathbf{w})\big)$ is strictly convex and monotonically increasing in $[0, 1]$, we have for all i:

$$\begin{cases} h_i(w_i) > -b\mu_i,\ h_i'(w_i) < 0, & \forall w_i \in [0,1] \\ \nu_i(\mathbf{w}) > 0,\ \frac{\partial \nu_i(\mathbf{w})}{\partial w_i} > 0, & \forall \mathbf{w} : \mathbf{0} \preceq \mathbf{w} \preceq \mathbf{1}. \end{cases} \tag{34}$$

All three cases in Fig. 2 satisfy (34). In Fig. 2(a), $\{w_1^{(t)}\}$ always converges to w_1^* for any initial value $w_1^{(0)} \in [0, 1]$, whereas in

Fig. 2(b), $\{w_1^{(t)}\}$ always diverges regardless of its initial starting point. In Fig. 2(c), however, the behavior of $\{w_1^{(t)}\}$ critically depends on its initial value $w_1^{(0)}$. If $w_1^{(0)}$ takes "initial value 1", $w_1^{(t)}$ will eventually hop between two values alternately, without being able to approach w_1^*. On the other hand, if $w_1^{(0)}$ takes "initial value 2", $w_1^{(t)}$ will converge to w_1^*.

We now give a sufficient condition for the convergence of chaotic price update in Theorem 1.

Theorem 1. *If for each $i = 1, \ldots, N$, we have*

$$(1-\epsilon)\big|\overline{U}_i''(w_i)\big| > \frac{\partial^2 c\big(K(\mathbf{w})\big)}{\partial w_i^2}, \tag{35}$$

for all $\mathbf{w}$ between $\mathbf{w}^{(0)} = \mathbf{1}$ and $\mathbf{w}^{(1)}$, then using chaotic price update, $k_i^{(t)}$ converges to k_i^* and $w_i^{(t)}$ converges to w_i^*.

Proof: Please refer to our technical report [15] for the proof. □

The economic implication behind Theorem 1 is that the algorithm will converge when *the marginal utility gain decreases faster than the marginal cost increases*, as w_i increases. This technical assumption can be easily justified, since the marginal cost $c'(K) = \beta$ for adding network capacity (routers and switches) is decreasing at a fast pace in our economy.

From the Proof of Theorem 1 in our technical report [15], the convergence speed of $\{k_i^{(t)}\}$ in chaotic price update is dictated by $P_i(w_i^*)$ and $Q_i(w_i^*)$, which depend on $h_i(w_i)$ and $\nu_i(\mathbf{w})$, the marginal utility gain and marginal cost in terms of w_i. Intuitively speaking, the larger the gap between the rates at which the marginal utility gain decreases and the marginal cost increases, the faster the convergence speed. As a result, in systems where $\big|\overline{U}_i''(w_i)\big|$ exceeds $\partial^2 c\big(K(\mathbf{w})\big)/\partial w_i^2$ by a substantial margin, the step-size-oblivious chaotic price update can achieve extremely fast convergence.

5.2 The Cutting-Plane Method

Chaotic price update achieves fast convergence when condition (35) is met. A natural question arises: *Can we design an algorithm that is step-size-free while converging under a wider range of conditions*? Now we present such an algorithm that converges for arbitrary concave utility function $\overline{U}_i(w_i)$ and arbitrary convex cost function $c\big(K(\mathbf{w})\big)$, with a *guaranteed* convergence speed. Our basic idea is to apply the *cutting-plane method* [8] to the dual problem of social welfare maximization, leading to an alternative formulation of the optimal pricing problem.

5.2.1 Dual Problem of Social Welfare Maximization

We introduce the dual problem of social welfare maximization, following the framework of consistency pricing [10, 21]. Note that the social welfare maximization problem (8) can be rewritten as

$$\begin{array}{ll}\max_{\mathbf{w},\mathbf{v}} & (1-\epsilon)\sum_i \big(\overline{U}_i(w_i) - b w_i \mu_i\big) - c\big(K(\mathbf{v})\big) \\ \text{s.t.} & \mathbf{w} = \mathbf{v},\end{array} \tag{36}$$

where the auxiliary vector $\mathbf{v}$ is introduced to facilitate *dual decomposition*. To derive the dual problem of (36), we define the *Lagrangian*

$$\begin{aligned} L(\mathbf{w},\mathbf{v},\mathbf{k}) &= (1-\epsilon)\sum_i \big(\overline{U}_i(w_i) - b w_i\mu_i\big) - c\big(K(\mathbf{v})\big) \\ &\quad + \mathbf{k}^{\mathsf{T}}(\mathbf{v}-\mathbf{w}) \\ &= (1-\epsilon)\sum_i \big(\overline{U}_i(w_i) - b w_i\mu_i\big) - \sum_i k_i w_i \\ &\quad + \sum_i k_i v_i - c\big(K(\mathbf{v})\big). \end{aligned} \tag{37}$$

Here k_i is the *Lagrange multiplier* associated with the ith equality constraint $w_i = v_i$; k_i can be interpreted as a *consistency price*, as it will eventually steer v_i towards w_i, as explained in [10, 21].

The *Lagrange dual function* is

$$q(\mathbf{k}) = \sup_{\mathbf{w},\mathbf{v}} L(\mathbf{w},\mathbf{v},\mathbf{k}), \tag{38}$$

and the *dual problem* of social welfare maximization (36) is

$$\min_{\mathbf{k}}\ q(\mathbf{k}). \tag{39}$$

Note that there is no duality gap between the dual problem (39) and primal problem (36) by the *strong duality theorem* [8], since the primal problem is convex optimization for any concave $\overline{U}_i(w_i)$ and convex $c\big(K(\mathbf{v})\big)$. As a result, it suffices to solve the dual problem instead of the primal problem.

In fact, the dual problem (39) is preferable because it enables distributed algorithms due to a natural decomposition of $q(\mathbf{k})$:

$$\begin{aligned} q(\mathbf{k}) = &\sum_i \sup_{w_i}\Big((1-\epsilon)\big(\overline{U}_i(w_i) - b w_i\mu_i\big) - k_i w_i\Big) \\ &+ \sup_{\mathbf{v}}\Big(\sum_i k_i v_i - c\big(K(\mathbf{v})\big)\Big). \end{aligned}$$

This dual decomposition "decouples" the objective function $q(\mathbf{k})$ so that the value of $q(\mathbf{k})$ can be found by solving a surplus maximization problem at each tenant i:

$$\max_{w_i}\ (1-\epsilon)\big(\overline{U}_i(w_i) - b w_i\mu_i\big) - k_i w_i, \quad \text{for all } i, \tag{40}$$

and a profit maximization problem at the cloud provider:

$$\max_{\mathbf{v}}\ \sum_i k_i v_i - c\left(K(\mathbf{v})\right). \tag{41}$$

As these subproblems are independent of each other, the dual problem enables distributed solutions by charging each tenant a reservation price k_i and usage price $p = b$ (the revenue and cost related to *usage* cancel each other in (41)). In contrast, the primal problem (36) is not decomposable because of the coupled term $c\left(K(\mathbf{v})\right)$.

5.2.2 Distributed Solutions via Cutting Planes

A traditional subgradient method will find a subgradient $\mathbf{g_k}$ of $q(\mathbf{k})$ at point $\mathbf{k}$, and update the prices $\mathbf{k}$ using this subgradient times a small step size. For example, one of such subgradients is given as below:

Lemma 1. *For any point $\mathbf{k} \in \mathbb{R}^N$, let the vector $\tilde{\mathbf{w}} = [\tilde{w}_i]$ be the optimal solutions to problem (40) and $\tilde{\mathbf{v}}$ be the optimal solution to problem (41). That is,*

$$\tilde{w}_i = \arg\max_{w_i}\ (1-\epsilon)\big(\overline{U}_i(w_i) - b w_i\mu_i\big) - k_i w_i, \quad \forall i, \tag{42}$$

$$\tilde{\mathbf{v}} = \arg\max_{\mathbf{v}}\ \sum_i k_i v_i - c\left(K(\mathbf{v})\right). \tag{43}$$

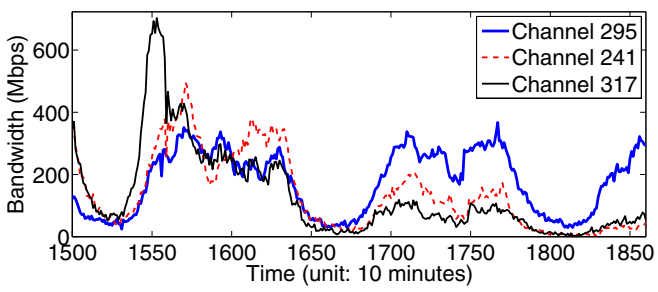
Then a subgradient of $q(\mathbf{k})$ at point $\mathbf{k}$ is given by $\mathbf{g_k} = \tilde{\mathbf{v}} - \tilde{\mathbf{w}}$.

Proof: Please refer to our technical report [15] for the proof. □

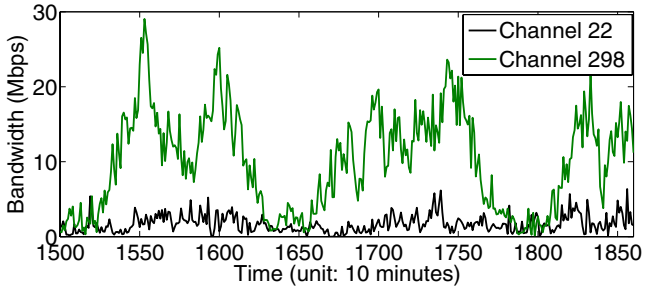
However, to ensure the convergence speed, the subgradient method requires tuning the step sizes strategically, which is difficult to implement in reality. In contrast, the cutting-plane method is step-size-free: it is essentially a *search algorithm* based on the following fact (see [8] for a proof):

Lemma 2. *Let vector $\mathbf{g_k} \in \mathbb{R}^N$ be a subgradient of the objective function $q(\mathbf{k})$ at point $\mathbf{k} \in \mathbb{R}^N$, i.e., $\mathbf{g_k}$ must satisfy*

$$q(\mathbf{x}) \geq q(\mathbf{k}) + \mathbf{g_k}^{\mathsf{T}}(\mathbf{x}-\mathbf{k}), \quad \forall \mathbf{x} \in \mathbb{R}^N. \tag{44}$$



(a) Three large channels



(b) Two small channels

Figure 3: Bandwidth consumption time series of 5 representative channels over a 2.5-day period.

Let $\mathbf{k}^*$ *be the optimal solution of the dual problem (39). Then* $\mathbf{k}^*$ *satisfies* $\mathbf{g}_{\mathbf{k}}^{\mathsf{T}}(\mathbf{k}^* - \mathbf{k}) \leq 0$.

Lemma 2 implies that if we know a point $\mathbf{k}$ with its subgradient $\mathbf{g}_{\mathbf{k}}$, we can confine our search for $\mathbf{k}^*$ within the half-space $\{\mathbf{x} : \mathbf{g}_{\mathbf{k}}^{\mathsf{T}}(\mathbf{x} - \mathbf{k}) \leq 0\}$, since the other half-space does not contain $\mathbf{k}^*$. Hence, we can locate $\mathbf{k}^*$ up to a certain accuracy by iteratively ruling out a sufficient number of half-spaces, as described below.

The Cutting-Plane Method. Set $\mathbf{k}^{(0)} = \mathbf{k}^{**}$, where $\mathbf{k}^{**}$ is an optimal solution for the non-multiplexing case. Set the initial polyhedron to be $\mathcal{P}_0 = \{\mathbf{k} | \mathbf{0} \preceq \mathbf{k} \preceq \mathbf{k}^{(0)}\}$. It is clear that $\mathcal{P}_0$ contains an optimal solution $\mathbf{k}^*$. For $t = 1, 2, \ldots$, repeat:

(1) Choose a point $\mathbf{k}^{(t)}$,which is the center of gravity of $\mathcal{P}_{t-1}$, denoted $\mathbf{k}^{(t)} = \text{CG}(\mathcal{P}_{t-1})$;

(2) Finding a subgradient $\mathbf{g}_{\mathbf{k}^{(t)}}$ of $q(\mathbf{k})$ at $\mathbf{k}^{(t)}$;

(3) If $\|\mathbf{g}_{\mathbf{k}^{(t)}}\|_\infty \leq \xi_1$, return $\mathbf{k}^{(t)}$; else, continue;

(4) Add a new cutting plane $\mathbf{g}_{\mathbf{k}^{(t)}}^{\mathsf{T}}(\mathbf{k} - \mathbf{k}^{(t)}) \leq 0$ to form the new polyhedron

$$\mathcal{P}_t := \mathcal{P}_{t-1} \cap \{\mathbf{k} | \mathbf{g}_{\mathbf{k}^{(t)}}^{\mathsf{T}}(\mathbf{k} - \mathbf{k}^{(t)}) \leq 0\}. \quad (45)$$

Intuitively speaking, the above algorithm attempts to shrink the volume of polyhedron $\mathcal{P}_t$ that contains the optimal solution $\mathbf{k}^*$ one iteration after another, until $\mathbf{k}^*$ is contained in a trivially small ball. In Step (2), the subgradient can be found using Lemma 2.

Now we can quantify the communication cost of the above algorithm — a crucial factor in a cloud environment. Since $\mathbf{k}^{(t)}$ is the center of gravity of $\mathcal{P}_{t-1}$, about half of the uncertainty is ruled out in each iteration. It can be proved that

$$\mathbf{vol}(\mathcal{P}_t) \leq \left(1 - \frac{1}{e}\right) \mathbf{vol}(\mathcal{P}_{t-1}) \approx 0.63 \cdot \mathbf{vol}(\mathcal{P}_{t-1}). \quad (46)$$

Therefore, the above algorithm converges *exponentially* fast. Furthermore, Lemma 2 shows that, in order to obtain a subgradient $\mathbf{g}_{\mathbf{k}^{(t)}}$ at $\mathbf{k}^{(t)}$, the cloud provider can simply charge each tenant a usage fee $p = b$ and a reservation fee $k_i = k_i^{(t)}$, and expect a return $\tilde{w}_i$ from each tenant i; it obtains $\tilde{\mathbf{v}}$ locally. Such price notification and response are performed in each iteration for all the tenants in parallel. In other words, each execution of Step (2) introduces only one *round* of message passing between the cloud and tenants. Since cloud-tenant communication happens only in Step (2), the total rounds of message-passing can be bounded as follows:

Proposition 2. *If the cutting-plane method terminates when the diameter of the smallest Euclidean ball that contains* $\mathcal{P}_t$ *is no greater than* d*, then in the worst case, the cutting-plane method requires*

$$R = 1.51N \log_2 \left(\frac{\max_i k_i^{(0)}}{d} \right) \quad (47)$$

rounds of message passing between the cloud and tenants.

The above proposition is a well-known property of the cutting-plane method [8]. In contrast, the worst-case communication cost of subgradient methods are of the form $O(N \times 1/\alpha)$, where α is the step size. Clearly, if d is comparable to α, the cutting-plane method may lead to much faster convergence due to the $\log_2(\cdot)$ operation.

It is worth noting that finding the center of gravity $\text{CG}(\mathcal{P}_{t-1})$ requires heavy computation. However, computation cost does not pose a challenge for data centers that have superior computing power, whereas communication cost (convergence speed) is the major bottleneck. The cutting-plane method converges faster, reducing the rounds of message passing between tenants and the cloud, yet at the expense of computational cost. Such a property is in fact desirable in the cloud. To summarize, the cutting-plane method converges for arbitrary concave utility $\overline{U}_i(w_i)$ and convex cost $c\big(K(\mathbf{w})\big)$, and can be used to compensate chaotic price update when the latter is not converging.

6. DEMAND STATISTICS ESTIMATION WITH A FACTOR MODEL

Recall that both algorithms for finding the optimal pricing policy need $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$ and $\boldsymbol{\Sigma}$ as inputs: μ_i and σ_i are used for the surplus maximization at each tenant i in (26) and (42), while the demand covariance matrix $\boldsymbol{\Sigma}$ is used to calculate the service cost $c(K(\mathbf{w}))$ in the presence of multiplexing in (27) and (43).

In this section, we address the practical issue of predicting demand statistics based on demand history, which can be obtained from cloud monitoring services such as Amazon CloudWatch [2] at a fine granularity (e.g., at a frequency of 5 minutes in CloudWatch). As has been mentioned, in this paper, we target applications whose bandwidth demand patterns are tractable and predictable to some extent. Video access is one example of such applications, with clear diurnal patterns and the time-of-the-day effect [17], in the sense that a popular video almost always sees its peak (or trough) demand around the same time of day.

Our study is based on a large dataset collected from thousands of on-demand video channels in a commercial VoD system [3] during the 2008 Summer Olympics. The video genres are not limited to Olympics, but range from TV episodes to sports and from movies

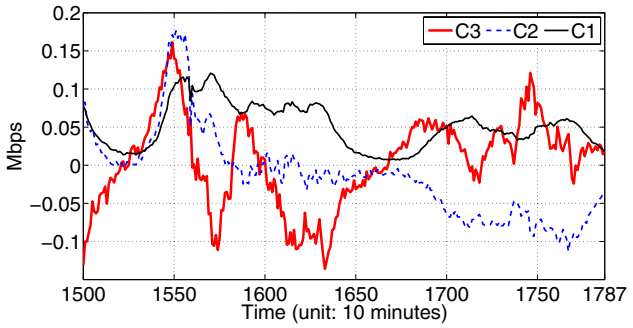


Figure 4: The first 3 principal components C_1—C_3 in bandwidth consumption series of 468 channels during 2 days via principal component analysis.

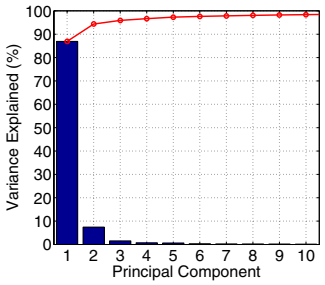


Figure 5: Variance explained in bandwidth consumption series of 468 channels in 2 days.

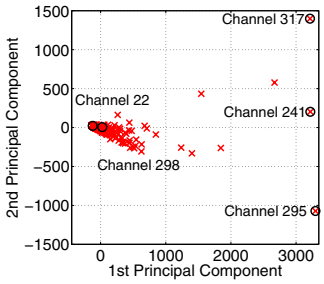


Figure 6: Data projected onto the first 2 principal components. Each point is one of the 468 channels.

to news. Fig. 3 shows the aggregate bandwidth demand in each channel for 5 representative channels over a 2.5-day period.

We have four observations about the dataset. *First*, the workload dataset consists of a large number of small unpopular channels, such as those in Fig. 3(b), dominated by a small number of large popular channels, such as those in Fig. 3(a). *Second*, there is a diurnal periodicity in the access pattern of each video channel. *Third*, a channel's popularity evolution may follow some trends over days. For example, bandwidth consumption in channels 241 and 317 exhibits a downward trend over the 2.5 days in Fig. 3(a), with 144 time periods representing one day. Finally, both the diurnal periodicity and daily trends become vague in small channels, such as in channel 22 in Fig. 3(b).

Our prior work has proposed to use seasonal ARIMA processes to predict bandwidth series in each individual channel [17] at a fine granularity of 10 minutes. The prediction is based on a regression of historical demand in the most recent time periods, as well as demand around the same time in previous days. However, this method has a shortcoming that a separate statistical model needs to be trained for every channel and thus does not scale to a large number of channels. Also, this approach performs poorly for small channels, e.g., channel 22, or ill-behaved channels, e.g., channel 317. In both types of channels, the daily repetition pattern is obscured by various random factors.

To tackle these problems, we use a *factor model* to account for demand evolutions, i.e., the demand series $\{D_i(t)\}$ of each channel i can be viewed as driven by M uncorrelated underlying factors $C_1(t), \ldots, C_M(t)$ with a zero-mean random shock $e(t)$:

$$D_i(t) = \alpha_{i1}C_1(t) + \ldots + \alpha_{iM}C_M(t) + e(t), \quad \forall i. \qquad (48)$$

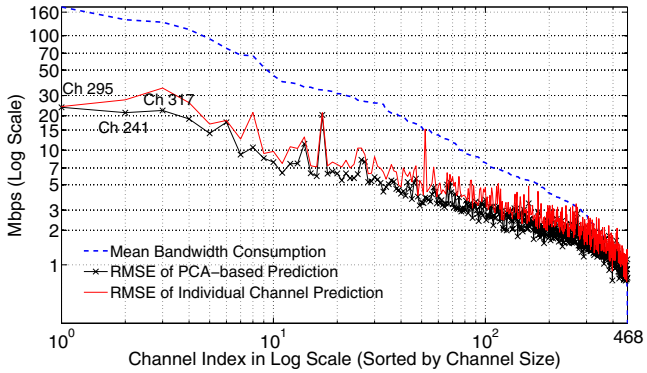


Figure 7: Root mean squared errors (RMSEs) of the two prediction methods over test period 1680-1860 (1.25 days), compared to mean bandwidth consumption in each channel. The three largest channels are channels 295, 241 and 317.

If the coefficients $\alpha_{i1}, \ldots, \alpha_{iM}$ can be learned statistically, we will be able to forecast $\{D_i(t)\}$ by predicting factor movements first. As noted from Fig. 3, channel demands exhibit co-movements. This inspires us to mine the factors while learning their coefficients from the collective demand history of *all* the channels.

We use principal component analysis (PCA) to find such underlying factors. Given N demand series $\{D_1(t)\}, \ldots, \{D_N(t)\}$, PCA applies an orthogonal transformation to these N demand series to obtain a small number of uncorrelated time series $\{C_1(t)\}$, $\ldots$, $\{C_M(t)\}$ called the *principal components*. This transformation is defined in such a way that data projected onto the first principal component has as high a variance as possible (that is, accounts for as much of the variability in data as possible), and each succeeding component in turn has the highest variance possible.

We perform PCA for all the 468 channels that are online in a 2-day period (time 1500-1787). Fig. 4 plots the first 3 principal component series in the data. We can see the first component C_1 explains the diurnal periodicity shared by all the channels. The second component C_2 accounts for the downward daily trend, which is salient in channel 317 as popularity diminishes and less salient in channel 295. A further check of Fig. 5 reveals that the first 10 principal component series explain 99% of the data variability, which are sufficient to model the factors underlying all the demand evolution.

However, different channels have different dependencies on each factor. Fig. 6 shows the 468 demand series projected onto the first 2 components, i.e., the point $(\alpha_{i1}, \alpha_{i2})$ for all i. Without surprise, the dependence on the first component, α_{i1}, indicates how large the channel is. In contrast, the dependence on the second component, α_{i2}, accounts for how fast end-users may lose interest in channel i. Channel 295 has a low α_{i2}, indicating almost no decrease in popularity over days. Channel 241 has a moderate α_{i2}, showing a slightly downward trend. Channel 317 has a large α_{i2}, exhibiting a dramatic decrease of popularity just on the second day.

To predict the demand means $\boldsymbol{\mu}(t)$ and covariances $\boldsymbol{\Sigma}(t)$ for $\{D_i(t) : i = 1, \ldots, N\}$ at time t, we first predict the principal components $C_1(t), \ldots, C_M(t)$ for $M = 10$ based on the history and obtain forecasts about their means $\hat{\mathbf{C}}(t) = [\hat{C}_1(t), \ldots, \hat{C}_M(t)]^{\mathsf{T}}$ and their covariances $\hat{\boldsymbol{\Sigma}}_C(t)$. We further predict the error series $e(t)$ to obtain its forecast $\hat{e}(t)$ and error variance $\hat{\sigma}_e^2(t)$. Note that $\hat{\boldsymbol{\Sigma}}_C(t)$ is a diagonal matrix because the principal components are uncorrelated. Denote the coefficient matrix as $\mathbf{A}_{N\times M} = [\alpha_{im}]$,

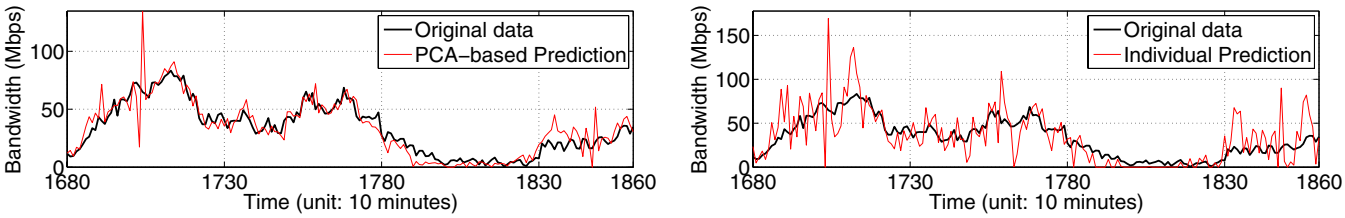
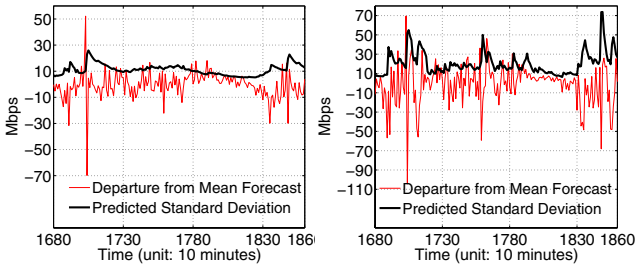
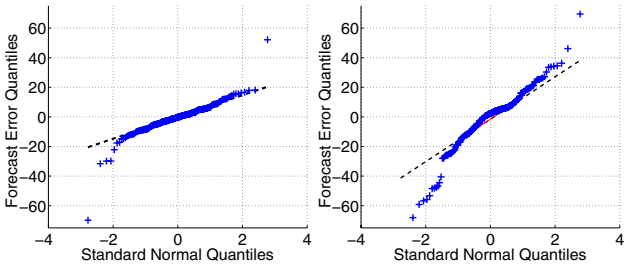


Figure 8: 10-minute-ahead conditional mean prediction in channel 172 over a test period of 1.25 days.



(a) PCA-based Prediction (b) Individual Prediction

Figure 9: The departure of actual bandwidth consumption from its conditional mean forecast and the predicted standard deviation of bandwidth consumption in channel 172.



(a) PCA-based Prediction (b) Individual Prediction

Figure 10: Q-Q plot of conditional mean forecast errors in channel 172 over the test period 1680-1860 (1.25 days), in reference to Gaussian quantiles.

$i = 1, \ldots, N$, $m = 1, \ldots, M$. We can therefore forecast $\boldsymbol{\mu}(t)$ and $\boldsymbol{\Sigma}(t)$ as

$$\hat{\boldsymbol{\mu}}(t) = \mathbf{A}\hat{\mathbf{C}}(t) + \hat{e}(t) \cdot \mathbf{1}, \tag{49}$$

$$\hat{\boldsymbol{\Sigma}}(t) = \mathbf{A}\hat{\boldsymbol{\Sigma}}_C(t)\mathbf{A}^{\mathsf{T}} + \hat{\sigma}_e^2(t) \cdot [\mathbf{1}, \ldots, \mathbf{1}], \tag{50}$$

where $\mathbf{1}$ is an all-one column vector of length N and $[\mathbf{1}, \ldots, \mathbf{1}]$ is an all-one matrix of size $N \times N$.

We model each principal component series using a low-order seasonal AIRMA model [9]. Since $\{C_1(t)\}$ clearly shows daily periodicity, we model $\{C_1(t) - C_1(t-144)\}$ as an ARMA$(1,1)$ process, so that the forecast $\hat{C}_1(t)$ is regressed from both the previous value $C_1(t-1)$, the values one day before $C_1(t-144)$, $C_1(t-145)$, and random noise terms. All other components $\{C_i(t)\}$ for $i \geq 2$ do not exhibit periodicity. We thus use ARMA$(1,1)$ processes to model these principal components. The conditional variances of all the component series are forecasted using GARCH$(1,1)$ models [9, 16]. Since the components are orthogonal, we do not need to forecast their covariances. For details of using seasonal AIRMA models and GARCH models for video traffic forecast, please refer to [16, 17].

We compare PCA-based prediction with individual channel prediction over a test period of 1.25 days. Each prediction is made based on the training data of only the previous 1.25 days, which are a little more than one day to incorporate periodicity. The root mean squared errors (RMSEs) of both approaches for all the 468 channels are summarized in Fig. 7. The channel indices are sorted in descending order of the channel size. We can see that the PCA-based approach outperforms individual predictions regardless of channel sizes. For large channels, the ratio of RMSE over mean bandwidth consumption is less than 15% in most cases using the PCA-based approach.

To zoom in, we take channel 172 as an example. Fig. 8 compares the conditional mean predictions produced by the PCA-based approach with those produced by individual prediction. We observe that individual prediction tends to oscillate drastically, while the PCA-based approach can better identify both periodicity and downward trends. One reason is that the driving factors found by PCA are weighted averages over all the channels, with channel-specific erratic noises smoothed out, exhibiting co-movements of all the channels.

The standard deviation forecast of channel 172 is plotted in Fig. 9. Even though there is a big gap between real demand and its conditional mean forecast around time 1700, the GARCH$(1,1)$ model is able to forecast a larger demand variance at this time, which will be leveraged by the cloud to allocate more capacity to guard against performance risks, using the technologies in Sec. 4.2. It is worth noting that we do not assume that demand can always be perfectly forecasted: the entire point of variance or volatility forecast via GARCH is to estimate the deviation of actual demand from the conditional mean prediction and enable risk management in a probabilistic sense. Fig. 10 shows the Q-Q plot of forecast errors. We observe that with PCA, the actual demand will oscillate around its conditional mean forecast more like a Gaussian process. This also substantiates the belief that each D_i behaves like a Gaussian random variable.

Last but not least, the PCA-based approach has a lower complexity: it involves training a seasonal ARIMA model for each of the 10 principal components, together with finding these components from the 468 channels using PCA. Once the models are trained, forecasting is simply a linear regression with negligible running time. In contrast, individual prediction has to train a seasonal ARIMA model for each of the 468 channels separately, leading to a much higher complexity.

7. TRADING SIMULATIONS

In this section, we simulate a computerized bandwidth reservation and trading environment based on our proposed algorithms. The simulation operates in rounds of 10 minutes. Before the start

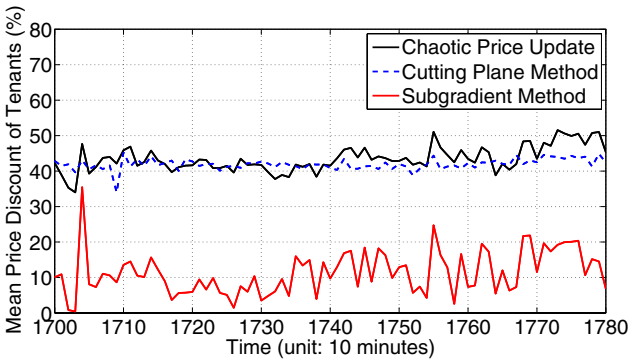


Figure 11: The mean price discount of all 100 tenants vs. time.

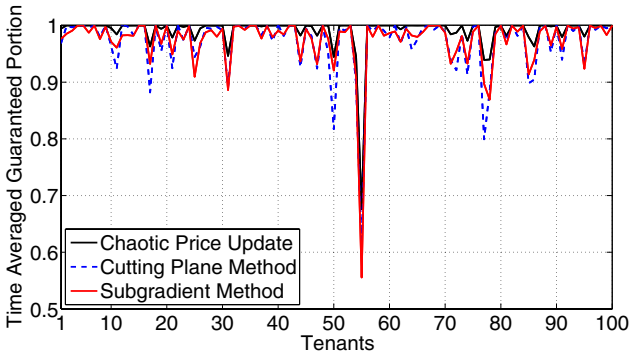


Figure 12: The guaranteed portion of each tenant averaged over all test periods 1700-1780.

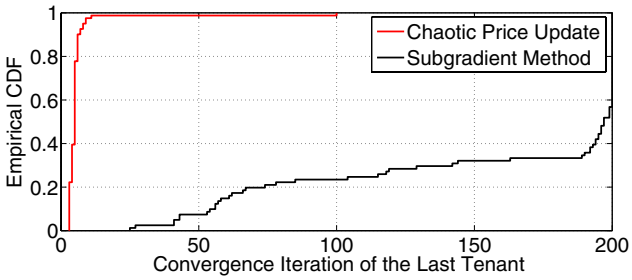


Figure 13: The CDF of the maximum convergence iteration of all tenants in each test period.

of each 10-minute period, the cloud provider has predicted the demand mean and covariances in this period and informed each tenant i about its specific μ_i and σ_i. When the period starts, the distributed price negotiation process immediately starts until convergence. Since in our particular problem, the cloud provider has superior computation power (even for finding polyhedra centroid), the delay is mainly due to the iterative message passing of prices and guaranteed portions between tenants and the cloud. We compare three algorithms: chaotic price update, the cutting-plane method and subgradient method, in terms of the convergence speed and optimization accuracy.

We consider 100 video channels of different sizes and statistics in the UUSee demand traces over a test period of 810 minutes. We assume each channel is a *tenant* that relies on the cloud for servicing the video requests from its end-users. We input such demand traces to our pricing framework and check the algorithm efficiency in the challenging case that prediction and optimization are to be carried out every 10 minutes. If the algorithms work for a 10-minute frequency, they will be competent for lower operating frequencies, such as on an hourly basis. We consider utility functions of the form (2). Under a Gaussian approximation of D_i, each tenant will have an expected utility

$$\mathbf{E}[U_i(w_i)] = \alpha_i w_i \mu_i - e^{A_i(1-w_i)\mu_i + \frac{1}{2}A_i^2(1-w_i)^2\sigma_i^2}. \quad (51)$$

The first term on the righthand side corresponds to the expected revenue of each tenant made from serving the demand $w_i D_i$, while the second term models a reputation loss which is convex and increasing in terms of the unfulfilled demand. In our simulation, we set $\alpha_i = 1$ and $A_i = 0.5$. Since different tenants have different μ_i and σ_i, their utilities are heterogeneous. We set the marginal cost of allocating bandwidth capacity to be $\beta := c'(K) = 0.5$, and assume that the cloud provider has an outage probability of $\epsilon = 0.01$.

We set the algorithm termination conditions as follows. In each iteration, if the change in either w_i, k_i or g_i ($\mathbf{g}_{\mathbf{k}^{(t)}} = [g_1^{(t)}, \ldots, g_N^{(t)}]$) is below some threshold, its w_i is not updated (using message-passing). Chaotic price update will stop if $|w_i^{(t)} - w_i^{t-1}| < 0.01$ or it has run for 100 iterations. The cutting-plane method will stop if $|k_i^{(t)} - k_i^{t-1}| < 0.05$ or it has run for 100 iterations. The subgradient method, as the benchmark, will stop if $|g_i^{(t)} - g_i^{(t-1)}| < 0.05$. In order to be generous to the benchmark algorithm, we set the maximum number of iterations for the subgradient method to 200. Note that in the subgradient method, the step size of price updates cannot be too small, which incurs slow convergence; it cannot be too big either, in which case the final output will be far away from the real optimal value. We optimize such a step size and set it to 0.1 for price updates. The other two algorithms are step-size-free.

We first compare the algorithm outputs upon termination. Note that with multiplexing, the final optimal price k_i^* for each tenant i should be lower than its initial value $k_i^{(0)} = \beta\big(\mu_i + \theta(\epsilon)\sigma_i\big)$, which is also the optimal price without multiplexing. We define $1 - k_i^*/k_i^{(0)}$ as the price discount that tenant i enjoys from multiplexing. Fig. 11 plots the mean price discount averaged over all the tenants in each test period. We observe that both chaotic price update and the cutting-plane method bring more discounts to tenants than the subgradient method. We further check the mean guaranteed portion chosen by each tenant averaged over all test periods in Fig. 12, which shows that most tenants choose a guaranteed portion close to 1 and the three algorithms are close to each other. This means although the three algorithms may reach a similar level of social welfare, the subgradient algorithm is not so good at fine-tuning the optimal prices for tenants with a guaranteed portion close to 1.

Finally, we check the communication overhead of all three algorithms. We define an iteration of message passing as a round-trip communication in which the cloud provider passes the prices to a tenant, which returns a chosen guaranteed portion. We observe that for chaotic price update, the convergence iteration of the last tenant (worst-case convergence iteration) in each test period is almost always less than 10. The CDF of the worst-case convergence iteration of chaotic price update is plotted in Fig. 13. In the same figure, we can observe that 40% of the time, the subgradient method needs 200 rounds to converge, while 60% of the time, it converges between 25 and 200 rounds. The cutting plane method always takes 100 rounds to converge, which are half of the maximum rounds needed by the subgradient method.

Taking both performance and speed into consideration, chaotic price update largely outperforms the other two. The cutting-plane method can also achieve better performance than the subgradient method with better price discounts reached within 100 rounds. Therefore, a practical strategy is to use chaotic price update first, and if it does not converge in 20 rounds, switch to the cutting-plane or subgradient method.

8. CONCLUDING REMARKS

Current-generation cloud computing platforms do not provide bandwidth guarantees, impeding the cloud adoption by tenants running QoS sensitive applications. Recent advancements in datacenter engineering augment the cloud-tenant interface with bandwidth reservation enabled. As bandwidth reservation becomes technically feasible, new models are needed to price the bandwidth guarantees to compensate the pay-as-you-go model which only prices the usage. In this paper, we propose a guaranteed cloud service model, where each tenant does not have to estimate the absolute amount of bandwidth it needs to reserve—it simply specifies a percentage of its demand from end-users that it wishes to serve with guaranteed performance, which we call the guaranteed portion, while the rest of the demand will be served with best efforts as the current cloud providers do. The cloud provider will estimate tenant demands through workload analysis and guarantee the performance in a probabilistic sense. The above process is repeated in small periods such as hours or tens of minutes.

Our main contribution is to fairly price such guaranteed services in each period. In contrast to the uniform usage pricing model, we price bandwidth reservations heterogeneously for tenants based on their workload statistics such as burstiness and correlation. It turns out to be computational challenge to find the optimal prices that maximize the expected social welfare under demand uncertainty. To address this challenge, we propose two novel distributed algorithms based on iterative equation updates and cutting-plane methods, which are oblivious to the choice of step sizes. We also propose practical algorithms to predict demand statistics based on a factor model. Trace-driven simulations show that both algorithms achieve faster convergence and better performance than subgradient methods. Given the abundant computing power and workload data in the cloud, our bandwidth reservation and algorithmic pricing system operates effectively at a fine granularity of as small as 10 minutes.

9. REFERENCES

[1] Amazon Cluster Compute, 2011. http://aws.amazon.com/ec2/hpc- applications/.

[2] Amazon Web Services. http://aws.amazon.com/.

[3] UUSee Inc. [Online]. Available: http://www.uusee.com.

[4] Four Reasons We Choose Amazon's Cloud as Our Computing Platform. *The Netflix "Tech" Blog*, Dec. 14 2010.

[5] D. Applegate, A. Archer, V. G. S. Lee, and K. Ramakrishnan. Optimal Content Placement for a Large-Scale VoD System. In *Proc. ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, 2010.

[6] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. A View of Cloud Computing. *Communications of the ACM*, 53(4):50–58, 2010.

[7] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron. Towards Predictable Datacenter Networks. In *Proc. of SIGCOMM'11*, Toronto, ON, Canada, 2011.

[8] D. P. Bertsekas, A. Nedic, and A. E. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, 2003.

[9] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel. *Time Series Analysis: Forecasting and Control*. Wiley, 2008.

[10] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle. Layering as optimization decomposition: A mathematical theory of network architectures. *Proceedings of the IEEE*, 95(1):255 – 312, Jan. 2007.

[11] R. L. Grossman. The Case for Cloud Computing. *IT Professional*, 11(2):23–27, March-April 2009.

[12] C. Guo, G. Lu, H. J. Wang, S. Yang, C. Kong, P. Sun, W. Wu, and Y. Zhang. SecondNet: a Data Center Network Virtualization Architecture with Bandwidth Guarantees. In *Proc. of ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, 2010.

[13] G. Gürsun, M. Crovella, and I. Matta. Describing and Forecasting Video Access Patterns. In *Proc. of IEEE INFOCOM Mini-Conference*, 2011.

[14] D. Kossmann, T. Kraska, and S. Loesing. An Evaluation of Alternative Architectures for Transaction Processing in the Cloud. In *Proc. of International Conference on Management of Data (SIGMOD)*, 2010.

[15] D. Niu, C. Feng, and B. Li. Pricing Cloud Bandwidth Reservations under Demand Uncertainty. Technical report, University of Toronto, 2012 http://iqua.ece.toronto.edu/~bli/papers/dniu-priceBR12.pdf.

[16] D. Niu, B. Li, and S. Zhao. Understanding Demand Volatility in Large VoD Systems. In *Proc. of the 21st International workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, 2011.

[17] D. Niu, Z. Liu, B. Li, and S. Zhao. Demand Forecast and Performance Prediction in Peer-Assisted On-Demand Streaming Systems. In *Proc. of IEEE INFOCOM Mini-Conference*, 2011.

[18] D. Palomar and M. Chiang. A tutorial on decomposition methods for network utility maximization. *IEEE J. on Sel. Areas in Communications*, 24(8):1439 – 1451, Aug. 2006.

[19] J. C. Panzar and D. S. Sibley. Public Utility Pricing under Risk: The Case of Self-Rationing. *The American Economic Review*, 68(5):888–895, Dec. 1978.

[20] J. Schad, J. Dittrich, and J.-A. Quiane-Ruiz. Runtime Measurements in the Cloud: Observing, Analyzing, and Reducing Variance. In *Proc. of VLDB*, 2010.

[21] C. W. Tan, D. Palomar, and M. Chiang. Distributed optimization of coupled systems with applications to network utility maximization. In *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing*, Toulouse, France, May 2006.

[22] C. Wilson, H. Ballani, T. Karagiannis, and A. Rowstron. Better Never Than Late: Meeting Deadlines in Datacenter Networks . In *Proc. of SIGCOMM*, 2011.

[23] C. Wu, B. Li, and S. Zhao. Multi-Channel Live P2P Streaming: Refocusing on Servers. In *Proc. of IEEE INFOCOM*, 2008.

[24] H. Yin, X. Liu, F. Qiu, N. Xia, C. Lin, H. Zhang, V. Sekar, and G. Min. Inside the Bird's Nest: Measurements of Large-Scale Live VoD from the 2008 Olympics. In *Proc. ACM Internet Measurement Conference (IMC)*, 2009.

[25] M. Zaharia, A. Konwinski, A. D. Joseph, Y. Katz, and I. Stoica. Improving MapReduce Performance in Heterogeneous Environments. In *Proc. of OSDI*, 2008.