A Study of Pricing for Cloud Resources

Hong Xu
Department of Electrical and Computer Engineering
University of Toronto
henryxu@eecg.toronto.edu

Baochun Li
Department of Electrical and Computer Engineering
University of Toronto
bli@eecg.toronto.edu

ABSTRACT

We present a study of pricing cloud resources in this position paper. Our objective is to explore and understand the interplay between economics and systems designs proposed by recent research. We develop a general model that captures the resource needs of various applications and usage pricing of cloud computing. We show that a uniform price does not suffer any revenue loss compared to first-order price discrimination. We then consider alternative strategies that a provider can use to improve revenue, including resource throttling and performance guarantees, enabled by recent technical developments. We prove that throttling achieves the maximum revenue at the expense of tenant surplus, while providing performance guarantees with an extra fee is a fairer solution for both parties. We further extend the model to incorporate the cost aspect of the problem, and the possibility of right-sizing capacity. We reveal another interesting insight that in some cases, instead of focusing on right-sizing, the provider should work on the demand and revenue side of the equation, and pricing is a more feasible and simpler solution. Our claims are evaluated through extensive trace-driven simulations with real-world workloads.

Keywords

Cloud computing, economics, pricing, throttling, performance guarantees, capacity right-sizing

1. INTRODUCTION

The abundant resources and flexible charging scheme of IaaS (Infrastructure as a Service) clouds have enabled the scalable and cost efficient deployment of numerous online services, such as storage backup (Dropbox), content delivery (Netflix), and application hosting (Yelp and Foursquare). In the cloud context, pricing is an important factor to the economics of the provider. On a related thread, many new systems designs have been proposed recently to improve the efficiency of operating a cloud [5, 8, 13, 18, 27].

The objective of this position paper is to understand the economical impact of current Industry practices and the economical potential of several new systems designs for the cloud, through a preliminary exploration of pricing. We study pricing of cloud resources for a monopoly provider operating an IaaS cloud with a fixed capacity. Pricing can be determined by a social welfare maximization problem in a competitive market with multiple providers, or by a provider revenue maximization problem in a monopoly market. This paper explores the latter and represents a starting point and a benchmark to studying the effect of pricing on resource and revenue management of an IaaS cloud.

We develop a microeconomic model similar to those used in congestion pricing for bandwidth allocation [7, 14, 17, 25] as the basis of our study in Sec. 2. We use canonical utility functions with varying utility levels to capture the resource needs of different applications for tenants. Usage pricing is adopted to model the predominant pay-as-you-go charging model for the provider. This is different from existing papers on bandwidth pricing that involve congestion externality in the utility functions and flat fee in the pricing model. Our model allows analyses to be tractable under a unified framework. As a preliminary study, many aspects of cloud resources and pricing are not captured in this model for tractability concerns, such as multi-resource requirements of applications [9, 10, 16], time-varying utility levels to model time-varying demand, and complicated utility functions to model the competition of multiple providers. We believe our model is general to consider these extensions in future work.

Our analyses reveal interesting insights to better understand common practices in the industry and assess the economic potential of some research directions in our community (Sec. 3). In the basic setting, we show that, a uniform usage price surprisingly does not suffer any revenue loss compared to first-order price discrimination, i.e. charging different tenants different prices. The provider can then safely restrict herself to developing a single usage price, which is the prevalent market practice today.

We then consider alternative strategies a provider can use to improve revenue, given that first-order price discrimination does not help. Since the cloud resources are of a best-effort nature, and the exact performance of a virtual machine is not explicitly specified, we consider *resource throttling* and its impact on pricing, which can be diligently exploited by the provider without much penalty today. We prove that throttling with pricing can achieve the optimal revenue as the tenant surplus is completely extracted. The provider thus has a significant financial incentive to throttle the resources of tenants. This result also provides a new perspective in explaining the severe performance degradation and variation of demanding applications in the cloud as widely reported in measurement studies [15, 22, 24, 31].

Continuing along the same line of thinking, the second strategy the provider can use to improve her revenue is to offer performance guarantees, a recent hot topic in the re-

search community [5, 13, 27]. We consider the revenue maximization problem where tenants are charged a premium in addition to the usage price for guaranteed resource allocation. We find that, tenants are able to retain positive surplus and do not suffer from performance degradation, and the provider can earn extra revenue compared to the basic setting. The recent efforts on providing guaranteed cloud resources [5, 13, 27] therefore have a much wider impact on the revenue and fairness issues of the cloud ecosystem, and need to be thoroughly explored.

We extend our framework to consider the operating cost of a cloud, and the possibility of right-sizing the capacity according to demand given recent developments [8, 18] in Sec. 4. Pricing is then determined by a profit maximization problem. We find that the optimal pricing has a threshold structure, and when the unit cost is small, prices should be set so that the capacity is fully utilized. This result implies that, though right-sizing helps save costs, in some cases, the provider should work on the demand and revenue side of the equation, and pricing proves to be beneficial and simple to implement for the efficient operation of the cloud. We use real-world workload traces to empirically evaluate the analyses in Sec. 5. We use three sets of traces from Google [11], the RIKEN Integrated Cluster of Clusters (RICC) in Japan [29], and the Intrepid cluster at Argonne National Laboratory (ANL) [28]. Each trace consists of at least tens of thousands of jobs run on thousands of nodes.

Finally, we would like to comment that, in cloud computing, pricing induces an intriguing interplay between systems and economics. This important angle should be explored by researchers, and we intend to touch upon several dimensions of such interplay and to provoke a broader discussion in this work as a first step. Our model and analyses are by no means complete. We do wish, however, that the issues brought up by our analyses, including resource throttling, performance guarantees, and capacity right-sizing, as well as others that we do not address, such as resource overprovisioning, encourage more thorough investigations especially from an economics perspective.

2. MODEL

In this section, we present our theoretical model for studying pricing in cloud computing.

The monopoly cloud we consider sells computational resources in the form of virtual machines at a price p. It adopts a pay-as-you-go charging model: A tenant that requires x virtual machines incurs a cost of px per unit time. We treat x as a real number instead of an integer for mathematical convenience. We consider a continuum of tenants. Each tenant attaches a certain utility function for its application running in the cloud. Similar to existing work on congestion pricing of bandwidth [7,14,17,25], we assume that the shape of the utility function is the same for all tenants, while the utility level varies across tenants. The varying utility level corresponds to the tenant's varying valuation for the application. Thus a tenant's utility function can be written as

$$vU(x), \tag{1}$$

where v denotes the utility level. Tenants with a utility level of v are called *type-v* tenants. In reality the cloud provider does not know the utility levels of individual tenants. Thus we assume a probabilistic model of utility levels, governed by the density function $f(v)$ defined over a range of $[v_0, v_1]$.

A rational tenant determines her demand for resources by solving an optimization that maximizes the difference between its utility and cost:

$$\max_x \ vU(x) - px. \tag{2}$$

Usually we adopt the canonical *alpha-fair* utility function [14, 20] to model applications' need for resources:

$$U(x) = (1-\alpha)^{-1} x^{1-\alpha}, \alpha \in (0,1). \tag{3}$$

With alpha-fair utility function, a tenant's demand function can be derived through the first-order condition of (2):

$$D_v(p) = \left(\frac{v}{p}\right)^{1/\alpha}. \tag{4}$$

This is the classical *iso-elastic* demand function widely used in economics [14, 19, 30]. The elasticity of demand is a standard measure of demand sensitivity to price changes and is defined as $-\frac{\mathrm{d}D_v \cdot p}{\mathrm{d}p \cdot D_v}$ [19]. For iso-elastic demand functions, elasticity equals $1/\alpha$, which is independent of utility level v and price p, and inversely proportional to α.

Finally, substituting the demand function into (2), a type-v tenant's *surplus* gained by using the cloud at price p, i.e. its optimal utility minus cost, is simply

$$S_v(p) = vU(D_v(p)) - pD_v(p) = \frac{\alpha p}{1-\alpha}\left(\frac{v}{p}\right)^{1/\alpha}. \tag{5}$$

3. PRICING FOR REVENUE MAXIMIZATION

In this section and Sec. 4, we present our pricing framework and analyses for cloud computing. We start by assuming that the operating costs are constant and pricing is determined from a revenue maximizing perspective. This is valid because the common practice is to leave all the servers of the cloud on and running all the time. We extend to consider the possibility of right-sizing the cloud capacity in Sec. 4, where pricing is then determined from a profit maximizing perspective.

3.1 Basic Setting with Uniform Usage Pricing

In practice, for reasons of simplicity and feasibility, the cloud provider usually adopts a uniform usage price for all tenants. She then needs to set an optimal price to maximize her revenue. The revenue obtained from a type-v tenant can be expressed as

$$R_v(p) = p \cdot D_v(p) = v^{1/\alpha} p^{1-1/\alpha}. \tag{6}$$

The revenue maximization problem with uniform pricing can then be defined as

$$\text{Basic\_OPT:} \quad \max \quad \int_{v_0}^{v_1} R_v(p) f(v) \mathrm{d}v \tag{7}$$

$$\text{s.t.} \quad \int_{v_0}^{v_1} D_v(p) f(v) \mathrm{d}v \le C, \tag{8}$$

$$S_v(p) \ge 0, \forall v, \tag{9}$$

$$\text{over} \quad p. \tag{10}$$

(8) corresponds to the resource capacity constraint, and C denotes the capacity. Since we consider a continuum of tenants, capacity here should be understood as capacity per

tenant. (9) is the individual rationality constraint that ensures tenants obtain non-negative surplus by utilizing the cloud resources.

However in theory, assuming the provider has complete information about tenant utility, charging a user-specific price, i.e. *first-order price discrimination* [19], can potentially improve revenue by extracting more surplus from higher valuing tenants. Though such a practice is deemed unfair by regulatory bodies [34], and almost infeasible to implement, it is important to characterize the potential revenue losses of uniform pricing as a starting point of pricing analyses.

With first-order price discrimination, a tenant is charged with a price p_v depending on her utility level v, and the revenue maximization problem is

$$\begin{aligned} \text{PD\_OPT:} \quad \max \quad & \int_{v_0}^{v_1} R_v(p_v) f(v) \mathrm{d}v \\ \text{s.t.} \quad & \int_{v_0}^{v_1} D_v(p_v) f(v) \mathrm{d}v \le C, \\ & S_v(p_v) \ge 0, \forall v, \\ \text{over} \quad & \{p_v\}. \end{aligned}$$

The provider has a set of prices $\{p_v\}$ to optimize instead of a single price p as in Basic_OPT.

We prove that, surprisingly, uniform pricing is not inferior to price discrimination in our model.

Theorem 1. *The revenue maximization problem with first-order price discrimination PD_OPT leads to the same solution as Basic_OPT with uniform pricing. The optimal pricing that maximizes the revenue in Basic_OPT and PD_OPT is given by the following:*

$$\begin{aligned} p^* &= \left(\frac{B}{C}\right)^{\alpha} = p_v^*, \forall v, \\ R^* &= B^{\alpha} C^{1-\alpha}, \\ S^* &= \frac{\alpha}{1-\alpha} B^{\alpha} C^{1-\alpha}, \end{aligned}$$

where the constant B is defined below, and R^* and S^* denote the optimal revenue and surplus, respectively.

$$B = \int_{v_0}^{v_1} v^{1/\alpha} f(v) \mathrm{d}v. \tag{11}$$

The proof is in Appendix A. Essentially, we show that the usage price depends on the utility level distribution and the elasticity parameter α, which are uniform across all tenants. Thus in cloud computing with usage pricing, the complicated first-order price discrimination does not offer an advantage, and a provider can safely restrict herself to using the simple uniform pricing.

Note that our result is in contrast to some existing studies that report significant revenue loss of uniform pricing [7, 14, 17]. The discrepancy comes from the specific model used. These works consider bandwidth pricing, where pricing strategies such as flat fees and utility functions with congestion externality are important. These factors do not properly model an IaaS cloud: Usage pricing is arguably the exclusive form of pricing used in practice, and congestion on CPU and memory resources in a cloud is *local* (per server) and does not have a network effect. Our result holds when the model considers the specifics of a cloud that involves only usage pricing and alpha-fair utility functions without congestion externality.

Examined more closely, Theorem 1 embraces some natural economical interpretations. First, price decreases when the cloud expands its capacity in order to attract more demand. When tenants are more sensitive to price, i.e. when α is smaller as discussed in Sec. 2, the price reduction is correspondingly smaller. Second, the optimal revenue increases with capacity *sublinearly* at $O(C^{1-\alpha})$. This result helps the cloud provider to make the optimal capacity planning decision given that the provisioning and operating cost is typically a convex or linear function of capacity [6, 18]. More discussions on this is deferred to Sec. 4. Third, tenants obtain a positive surplus by using the cloud, which also grows sublinearly with capacity.

Observe that the tenant surplus $S_v(p)$ in (5) is always non-zero for all prices, from the provider's perspective this implies that the basic usage pricing is not efficient enough to extract all possible revenues. Since we have shown that first-order price discrimination does not help, in the following sections we consider other unique aspects of cloud computing that can help the provider improve her revenue together with intelligent pricing.

3.2 Pricing with Resource Throttling

Most cloud providers today do not have a comprehensive Service Level Aggrement (SLA) guaranteeing the performance and reliability of the service. Thus, although the hardware configuration is explicit, the exact performance of a virtual machine, such as its I/O speed, network bandwidth, is not specified. For example Amazon EC2 abstracts the CPU resource in terms of "EC2 compute unit," with one compute unit equivalent to a 1.0–1.2 GHz 2007 Xeon processor [1]. The I/O performance is also vaguely defined as "high," "moderate," and "low" without clear explanations [32].

The best-effort nature of the current cloud offering leaves enough room for the provider to diligently *throttle* the resources of their virtual machines without being penalized. Technically, CPU and I/O throttling can be done by adjusting the scheduling weight of a virtual machine and limiting the maximum amount of resources it can consume through the hypervisor [2]. The provider can also vary the degree of throttling over time to mask the effect. Intuitively throttling slows down applications and forces tenants to buy more resources. Indeed, many measurement studies report severe computational performance degradation and variation of public clouds for HPC applications [15, 22, 24, 31]. This motivates us to investigate the potential impact of throttling on revenue and pricing.

We observe that tenants understand they are using a best-effort service, and tolerate performance degradation and variation. Also it is difficult to measure the exact degree of throttling due to performance variation. Thus the tenant demand function does not change with performance throttling. Throttling can be modeled by the provider offering a fraction $\beta \in (0, 1)$ of the required resources to tenants, where β denotes the slowdown factor. Effectively, a tenant only obtains $\beta D_v(p)$ resources, and her surplus function becomes

$$S_v(p, \beta) = vU(\beta D_v(p)) - pD_v(p) = p\left(\frac{v}{p}\right)^{1/\alpha}\left(\frac{\beta^{1-\alpha}}{1-\alpha} - 1\right) \tag{12}$$

When no throttling is used, i.e. $\beta = 1$, the above reduces to the surplus function in the basic setting (5). Thus the

revenue maximization problem with throttling can be formulated as

$$\begin{aligned} \text{Throttling\_OPT:} \quad \max \quad & \int_{v_0}^{v_1} R_v(p) f(v) \mathrm{d}v \qquad (13) \\ \text{s.t.} \quad & \int_{v_0}^{v_1} \beta D_v(p) f(v) \mathrm{d}v \le C, \\ & S_v(p,\beta) \ge 0, \forall v, \\ \text{over} \quad & p, \beta. \end{aligned}$$

Compared to Basic_OPT, now the provider can adjust both the slowdown factor β and the price to optimize revenue.

Observe from (12) that when the surplus is positive, we can always decrease the slowdown factor β to increase the effective capacity of the cloud. Price will be reduced with capacity expansion, and revenue can be improved as $R_v(p)$ is decreasing in p. Therefore the maximum must be achieved when the individual rationality constraint is satisfied at equality, and we can prove the following:

Theorem 2. *The optimal pricing that maximizes the revenue in Throttling_OPT is given by the following:*

$$\begin{aligned} \beta^* &= (1-\alpha)^{1/1-\alpha} < 1, \\ p_t^* &= (1-\alpha)^{\alpha/1-\alpha} \left(\frac{B}{C}\right)^{\alpha}, \\ R_t^* &= \frac{B^\alpha C^{1-\alpha}}{1-\alpha}, \\ S_t^* &= 0, \end{aligned}$$

and compared with the basic results in Theorem 1, the price reduction and revenue improvement are:

$$\begin{aligned} \frac{p_t^*}{p^*} &= (1-\alpha)^{\alpha/1-\alpha}, \\ \frac{R_t^*}{R^*} &= \frac{1}{1-\alpha}. \end{aligned}$$

Thus, the use of throttling actually enables the provider to achieve the *maximum* revenue theoretically possible, because all of the tenant surplus is extracted. Price can be reduced by a factor of $(\beta^*)^{-\alpha}$ to encourage more demand to utilize the additional capacity saved by throttling. Revenue can be increased by a factor of $1/1-\alpha$, and the total surplus is drained to zero. The advantages of throttling comes from the concavity of the utility functions, i.e. $\alpha > 0$. Concavity is in general satisfied in practice because the marginal utility gain of receiving more resources is diminishing for tenants. Thus, in general, the cloud provider has a strong financial incentive to exploit the best-effort nature of cloud offering and throttle the performance of virtual machines in order to increase her own revenue, at the expense of tenants.

Notice that throttling is not easy to achieve for ISPs selling Internet bandwidth, which also has a best-effort nature. There is only one kind of resources involved with explicit specification (maximum downlink/uplink speed), and the amount of bandwidth can be easily measured. The mature market with numerous ISPs also imposes significant penalty of using throttling. The cloud market does not possess these qualities.

3.3 Pricing with Performance Guarantees

A best-effort cloud makes tenants vulnerable to exploitation of the provider as we have already seen. Moreover, some tenants may run mission-critical jobs and require strict performance guarantees, and important applications such as distributed large-scale Mapreduce and Hadoop jobs do not perform well with best-effort resources. As a result, recently a large amount of efforts have been made to technically enable a cloud to provide guaranteed services [5, 13, 27].

Unlike throttling that gives the provider an unfair advantage to exploit tenants, performance guarantees can be mutually beneficial. On one hand tenants enjoy consistent and predictable performance. On the other hand the provider can also earn extra revenue by charging a premium for the guaranteed service. Here we study this new pricing problem to understand the exact tradeoff achieved between tenant surplus and provider revenue.

In this case, the provider explicitly guarantees the resources allocated to a tenant (i.e., no throttling) at all times, and charges an additional usage independent premium q for the SLA. The automation of monitoring and technical support largely explains the resource independent nature of the SLA charge. Thus the demand function does not change with the SLA charge q. The surplus function has an additional component of $-q$:

$$S_v(p,q) = vU(D_v(p)) - pD_v(p) - q = \frac{\alpha p}{1-\alpha}\left(\frac{v}{p}\right)^{1/\alpha} - q, \qquad (14)$$

and the revenue function becomes

$$R_v(p,q) = p \cdot D_v(p) + q = v^{1/\alpha} p^{1-1/\alpha} + q. \qquad (15)$$

The revenue maximization problem with the SLA charge can be formulated:

$$\begin{aligned} \text{SLA\_OPT:} \quad \max \quad & \int_{v_0}^{v_1} R_v(p,q) f(v) \mathrm{d}v \qquad (16) \\ \text{s.t.} \quad & \int_{v_0}^{v_1} D_v(p) f(v) \mathrm{d}v \le C, \\ & S_v(p,q) \ge 0, \forall v, \\ \text{over} \quad & p, q. \end{aligned}$$

Now the provider can adjust both the SLA charge q and the usage price p to maximize revenue.

Theorem 3. *The optimal pricing that maximizes the revenue in SLA_OPT is given by the following:*

$$\begin{aligned} q^* &= \frac{\alpha v_0^{1/\alpha}}{1-\alpha}\left(\frac{B}{C}\right)^{\alpha-1}, \\ p_s^* &= \left(\frac{B}{C}\right)^{\alpha} = p^*, \\ R_s^* &= B^\alpha C^{1-\alpha}\left(1 + \frac{\alpha v_0^{1/\alpha}}{(1-\alpha)B}\right), \\ S_s^* &= \frac{\alpha}{1-\alpha} B^\alpha C^{1-\alpha}\left(1 - \frac{v_0^{1/\alpha}}{B}\right), \end{aligned}$$

and compared with the basic results in Theorem 1, the revenue improvement and surplus reduction are:

$$\begin{aligned} \frac{R_s^*}{R^*} &= 1 + \frac{\alpha v_0^{1/\alpha}}{(1-\alpha)B}, \\ \frac{S_s^*}{S^*} &= 1 - \frac{v_0^{1/\alpha}}{B}. \end{aligned}$$

Moreover, we can show the following:

Lemma 1. *$R_s^* < R_t^*, S_s^* > S_t^*$. The optimal revenue with a SLA charge is less than that with throttling, while the optimal surplus is more than that with throttling.*

The proofs can be found in Appendix B. Thus, the tenants are better off using the guaranteed service with an extra charge, because throttling will drain the surplus to zero for the best-effort service. The provider cannot improve her revenue as much as she could with throttling. The revenue improvement depends on the spread of utility levels as capture by the term $v_0^{1/\alpha}B$. If the probability mass is concentrated around the lowest utility levels v_0, the revenue improvement ratio is large. If the probability is spread out across a large range of values, $v_0^{1/\alpha}B$ becomes a small number, and the revenue improvement ratio is also small. However, with the scale of a cloud, even a small percentage of revenue improvement translates to significant monetary returns as we shall demonstrate in Sec. 5.4. Overall, offering performance guarantees with an extra charge is a more viable and fairer solution for both parties.

The reason for the performance discrepancy is that throttling, with the slowdown factor β, has a *multiplicative* effect on tenant utility, independent of the utility level v, and by optimizing β the provider can extract all the tenant surplus. The optimization (13) is solved when $S_v(p,\beta) = 0$ for all utility levels v. The SLA charge q only has an *additive* effect on tenant utility, which also depends on the utility level. A uniform SLA charge helps reduce the tenant surplus and improve revenue, but not to the extent that tenants have zero surplus. The optimization (16) is solved when $S_v(p,q) = 0$ for only tenants with the lowest utility level v_0.

4. PRICING FOR PROFIT MAXIMIZATION: OPERATING COST AND CAPACITY RIGHT-SIZING

So far we have considered the scenario when the cloud capacity is pre-determined and always available, and optimized pricing for revenue maximization. While this corresponds to the current industry practice that always leaves servers on [12], a large body of work has been devoted to *right-sizing* the capacity of the cloud, i.e., turning down abundant servers (c.f. [8, 18] and references therein) and switches [26]. In this section, we extend our model to consider right-sizing and the cost aspect of running the cloud, and study its effect on the pricing decision.

With right-sizing, the capacity can be tuned by the provider (up to C) according to the demand. The operating costs of the cloud then can be modeled as a convex function of the demand. The convexity assumption is general and captures many common server models. One example is to say that the energy cost of servers and switches constitutes a majority of the operating costs [12], which is often modeled using an affine function [6, 18]:

$$E(x) = e_0 + e_1 x. \tag{17}$$

x is the total demand. e_0 models the fixed energy costs independent of workload, and e_1 is the variable energy cost per unit of resources. In practice, we expect that $c(\cdot)$ will be empirically measured by observing the system over time.

Substitute $x = D_v(p)$ into $E(x)$, we obtain the cost function with respect to p for a type-v tenant:

$$E_v(p) = e_0 + e_1 v^{1/\alpha} p^{-1/\alpha} \tag{18}$$

$$\begin{aligned} \text{Cost\_OPT:} \quad \max \quad & \int_{v_0}^{v_1} (R_v(p) - E_v(p)) f(v)\mathrm{d}v \qquad (19)\\ \text{s.t.} \quad & \int_{v_0}^{v_1} D_v(p) f(v)\mathrm{d}v \le C, \\ & S_v(p) \ge 0, \forall v, \\ \text{over} \quad & p. \end{aligned}$$

The optimal pricing can be derived from the KKT conditions shown in Appendix C:

Theorem 4. *The optimal pricing that maximizes the profit in Cost_OPT is given by the following.*
When $e_1 > (1-\alpha)B^\alpha C^{-\alpha}$,

$$\begin{aligned} p_c^* &= \frac{e_1}{1-\alpha}, \\ D^* &= B\left(\frac{e_1}{1-\alpha}\right)^{-1/\alpha} < C, \\ P^* &= \alpha B\left(\frac{e_1}{1-\alpha}\right)^{1-1/\alpha} - e_0, \end{aligned}$$

when $e_1 \le (1-\alpha)B^\alpha C^{-\alpha}$,

$$\begin{aligned} p_c^* &= \left(\frac{B}{C}\right)^\alpha = p^*, \\ D^* &= C, \\ P^* &= B^\alpha C^{1-\alpha} - e_1 C - e_0, \end{aligned}$$

where D^* denotes the optimal demand at price p_c^*, and P^* is the maximum profit.

The implications of Theorem 4 are interesting and very important. The benefit of right-sizing, given that the price is set optimally, critically depends on the unit cost of running the cloud. When the unit cost e_1 is high, the provider should set the price so that the demand is smaller than capacity, and use right-sizing to maximize profit. If the unit cost is low, the price should be set so demand equals capacity. In this case the provider is better off leaving all the servers on, and adjusting the price to maximize capacity utilization and revenue.

We do not imply that the literature on right-sizing datacenters is ill-directed. Note that we consider the static case here, while for the dynamic case where demand changes over time, right-sizing surely helps the provider saving unnecessary costs. Nevertheless, our result still offers valuable insights in this case. In fact the insights of Theorem 4 can be better understood in the dynamic setting. Demand dynamics can be interpreted as a result of time-varying utility level v. During demand valley periods, v is low and B is small. When the unit cost is high, pricing does not help since the optimal price p_c^* does not change with v. Right-sizing should be adopted to maximize profit, and the end result is that the total demand and profit are lower.

However, when the unit cost is low, Theorem 4 says the optimal price p_c^* depends on B and v and is lower when v is smaller. In this scenario, the provider should work on the revenue side of the problem and lower the price at valley periods to use up her capacity and maximize profit, instead

Trace	Nodes	Jobs	Duration	Ave. & Max. CPU per job
Google	≈ 11K	1 million+	29 days	53.7947, 800
RICC	1024	447794	5 months	6.4714, 2048
ANL	40960	68936	5 months	41.8673, 7950

Table 1: Statistics of three traces used.

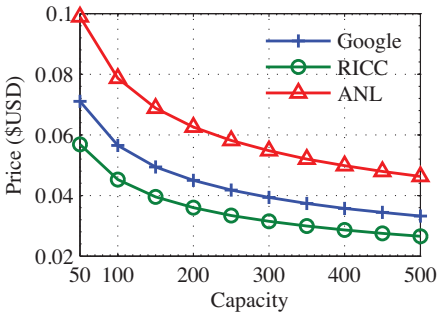

Figure 1: Optimal price p^* of Basic_OPT.

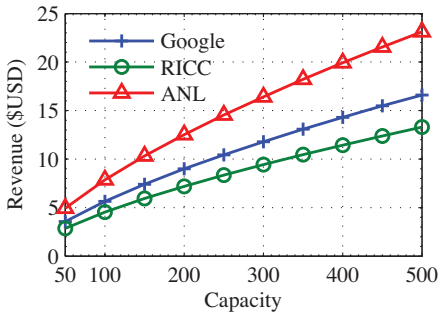

Figure 2: Optimal revenue R^* of Basic_OPT.

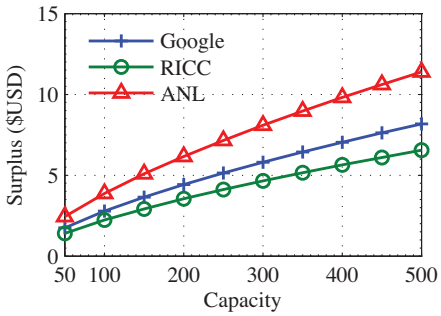

Figure 3: Optimal surplus S^* of Basic_OPT.

of working on the cost side and using right-sizing. This key message is resonant with the conclusion of [12] to leave all servers on and improve their utilization for a better profit. Pricing is more beneficial and easier to implement for the efficient operation of a cloud in this case, given the wear-and-tear costs of right-sizing and the technical difficulty involved [18].

Our discussion above also leads to *dynamic pricing* where prices may be adjusted over time when the tenant utility level changes. This serves as an interesting direction of our future work, and is beyond the scope of this paper. Also note that pricing with resource throttling and performance guarantees studied in Sec. 3 can be similarly analyzed in this profit maximization framework, which we omit due to space constraints.

5. EVALUATION

We present our evaluation results in this section.

5.1 Setup

One significant hurdle of conducting pricing and other studies in cloud computing is the lack of empirical data from real-world providers. We resolve this issue here by relying on empirical traces of workloads from large-scale computer clusters. Specifically, we use three traces from Google [11], RIKEN Integrated Cluster of Clusters (RICC) in Japan [29], and the Intrepid cluster at Argonne National Laboratory (ANL) [28]. The Google cluster mainly runs production jobs while the RICC and ANL clusters run scientific and engineering computing jobs. The statistics of the three traces are presented in Table 1. Though these traces are from private clusters instead of a production cloud, we believe the nature and scale of the workloads faithfully reflect those of applications running in a cloud, and the first-order estimation of cloud workloads is appropriate here.

The workload traces are used to calculate the empirical distribution of the utility level $f(v)$. The procedure is as follows. We process the traces to obtain the amount of resources (number of CPU cores) each job requires[1]. Assuming the workloads run in a cloud, and one CPU core is equivalent to one small linux instance in Amazon EC2 and Microsoft Azure, we then have the demand information $D_v(p)$ at the market price of $0.08/hour for a small linux instance in June 2012. Given the heated competition between public clouds, the elasticity parameter is set to $1/\alpha = 3$, i.e. $\alpha = 0.33$ to model the price sensitive nature of cloud resources. From (4), we can readily calculate the utility level v for individual jobs, and obtain the distributions of utility levels $f(v)$ from all three traces. For example, Figure 7 shows the utility level distribution from the Google trace.

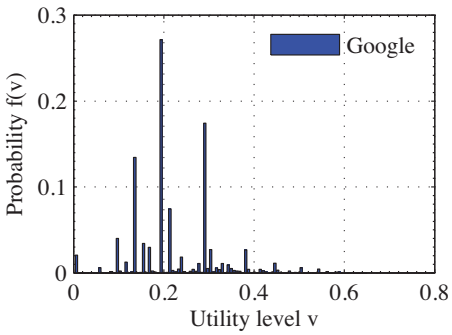

Figure 7: Utility level distribution $f(v)$ from the Google trace.

Most public clouds bill on an hourly basis. Thus in our evaluation the time unit is one hour, which is consistent with the procedure above using the market hourly price. All the metrics such as price and revenue shall be interpreted on a per hour per tenant basis in the following figures and paragraphs.

5.2 Baseline

[1]Since Google normalizes the trace, we scale the values by assuming the smallest CPU requested is equal to 1 core.

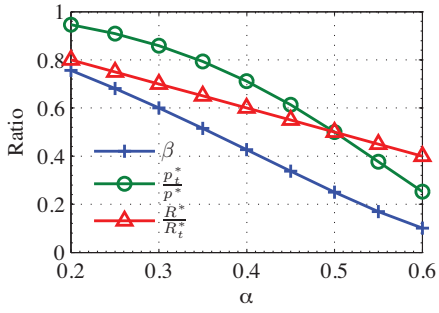

Figure 4: Optimal slowdown factor β^*, price reduction $\frac{p_t^*}{p^*}$, and inverse of revenue improvement $\frac{R^*}{R_t^*}$ of Throttling_OPT.

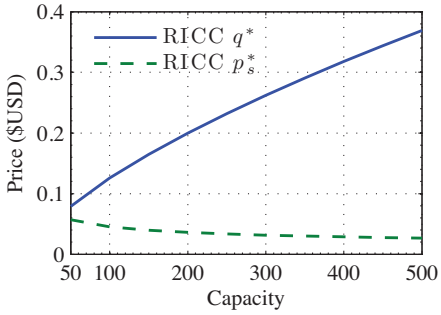

Figure 5: Optimal SLA charge q^* and price p_s^* of SLA_OPT for the RICC dataset.

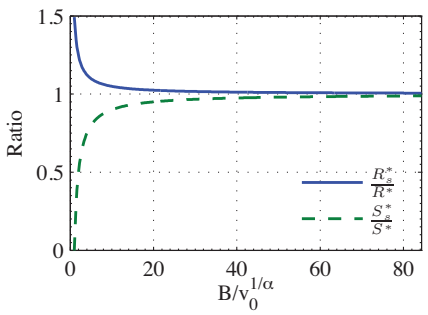

Figure 6: Optimal revenue improvement $\frac{R_s^*}{R^*}$ and surplus reduction $\frac{S_s^*}{S^*}$ of SLA_OPT.

We first solve the Basic_OPT with the three traces to evaluate the performance of usage pricing as the baseline of other pricing schemes. We vary the capacity per tenant C from 50 to 500. Figure 1 shows the optimal usage price p^*. The price ranges from \$0.1 to \$0.017, and is slowly decreasing with capacity. Figure 2 and 3 show the optimal revenue R^* and surplus and S^*, respectively. The sublinear growth rate is clearly demonstrated as proved in Theorem 1. Note that the optimal price, revenue and surplus obtained from the ANL trace is significantly larger than other two traces, because the ANL jobs are much larger due to their scientific computing nature.

We wish to point out the significant revenue potential of cloud computing. At an average capacity of 100 virtual machines for a tenant, the revenue is around \$5 per hour per tenant. If the cloud had 1000 tenants, the total revenue will be \$5000 per hour, or \$0.12 million per day. Though these numbers are only rough estimates and our model remains theoretical, the revenue potential partly explains the burgeoning of public cloud offerings today [3].

5.3 Throttling

We evaluate the pricing and revenue impact of throttling now. Based on Theorem 2, the slowdown factor β^* and performance improvements depend only on the elasticity parameters. Thus we plot the numerical values in Figure 4 with varying α. Be reminded that results of other sections are obtained when α is set to 0.33 as explained in Sec. 5.1.

Observe that β^* decreases with inverse elasticity α. When $\alpha = 0.33$, $\beta^* = 0.5501$, suggesting that the provider will throttle the VMs so tenants only obtain 55% of the achievable performance. The result may sound surprising. However, measurements report that the sustained computational performance of EC2 suffers more than 50% degradation compared to equivalent hardware [15, 31]. On the other hand, as the price elasticity of cloud resources becomes smaller (α becomes larger), more severe throttling is required to obtain the maximum revenue, which may not be feasible to implement in reality. In this case, the provider needs to consider the negative impact of throttling on demand in order to determine the optimal strategy.

The usage price reduction $\frac{p_t^*}{p^*}$ ranges from around 1 to around 0.2. The inverse of revenue improvement $\frac{R^*}{R_t^*}$ ranges from 0.8 to 0.4, meaning that the revenue improvement is from 25% to 150%. Again we stress that the numbers only serve as rough estimates. The key message is that throttling is a viable and financially attractive strategy that providers can use to improve revenue at the expense of tenants of a best-effort cloud.

Trace	R_s^*/R^*	R^* when $C = 100$	revenue improv.
Google	1.0190	\$5.1282	\$2338.5
RICC	1.0278	\$4.5205	\$3016.1
ANL	1.0195	\$7.9848	\$3641.1

Table 2: Optimal daily revenue improvement of SLA_OPT with 1000 tenants and $C = 100$.

5.4 SLA Charge

Next we evaluate the performance of the SLA charge together with usage pricing. In Figure 5, we plot the optimal SLA charge q^* with the usage price p_s^*, which is equal to p^* in the baseline case as shown in Theorem 3. For clarity of presentation we only present the results from the RICC trace. The SLA charge is increasing with capacity to extract the increasing tenant surplus, while the usage price is decreasing to attract demand and maximize capacity utilization. The resource usage independent SLA charge is also larger than the usage price in general.

Figure 6 shows the revenue and surplus ratio $\frac{R_s^*}{R^*}$ and $\frac{S_s^*}{S^*}$ compared to the baseline. A quick observation is that both depends heavily on the distribution of utility levels. When the distribution is geared towards small values, the ratio $B/v_0^{1/\alpha}$ is small, and revenue improvement is significant. When the distribution concentrates around large values, the revenue improvement becomes marginal. Thus the revenue impact of the SLA charge is not as significant as throttling.

However, recall that the revenue in dollar terms is quite significant for a large-scale cloud. Thus even a single digit percentage improvement amounts to sizable financial returns. This point is illustrated in Table 5.4 that shows the absolute value of revenue improvement by SLA charge for a cloud with 1000 tenants and 100 virtual machines per tenant (i.e. $C = 100$). A daily revenue improvement of thousands of dollars can be expected.

5.5 Operating Cost

We finally evaluate the impact of the operating cost to

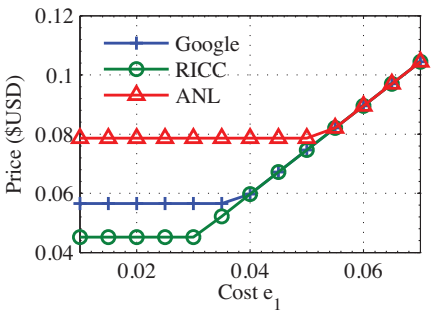

Figure 8: Optimal price p^* of Cost_OPT.

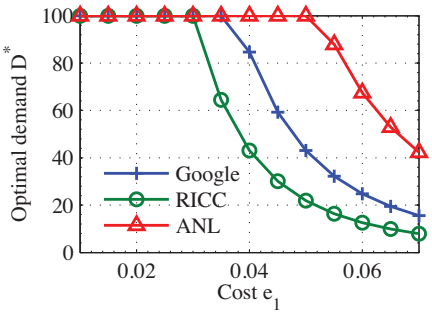

Figure 9: Optimal capacity C^* of Cost_OPT.

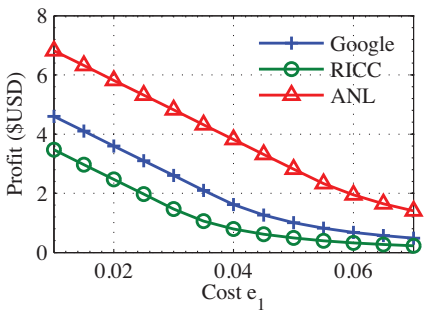

Figure 10: Optimal profit P^* of Cost_OPT.

the pricing problem. We solve the Cost_OPT for the three traces with the per unit cost e_1 varying from \$0.01 to \$0.07. The fixed cost e_0 is set to \$0.05, and the capacity is fixed at 100 here. Figure 8 and 9 show the optimal price p_c^* and demand D^* against e_1. Observe that the threshold structure as proved in Theorem 4 is clearly demonstrated. When the unit cost e_1 is small, p_c^* is set to maximize the capacity utilization, and does not change with cost. D^* is always equal to the capacity 100. Again, since different traces have different utility level distributions and thus different B, their optimal p_c^* is different. When the unit cost is high, prices should be set according to the cost to maintain a profit margin, and thus does not vary across traces. Demand then becomes less than the capacity and depends on the trace-specific term B. The overall effect as shown in Figure 10 is that the optimal profit P^* decreases when cost increases, which is intuitive to understand.

6. RELATED WORK

An extensive literature exists on pricing in communication networks and the Internet. The most related works are [7, 14, 17, 25]. [7, 17, 25] study the benefits of first-order price differentiation. Hande et al. [14] characterize the economic loss due to the ISP's inability or unwillingness to price broadband access based on time of day. The similarities and differences between these work and ours are discussed in Sec. 3.1. Other related works on bandwidth pricing include [30] that studies second-order price discrimination, i.e. tiered pricing based on usage volume and traffic destinations, to achieve service differentiation and congestion control. Tiered pricing based on volume is an interesting future direction to extend our general framework to.

There have been some recent studies on pricing of cloud resources. [33] argues for the importance of pricing in cloud computing for distributed systems design. From a user's perspective, [23] envisions a flat fee scheme for clouds and solves the optimal resource auto-scaling problem. From a provider's perspective, [4] studies the revenue impact of fixed and spot pricing in cloud computing, and [21] studies a pricing strategy specifically designed for bandwidth resources of a cloud based VoD system. Our work does not focus on designing a better pricing scheme. Instead, through pricing analyses, it tries to explore the economical impact of several unique systems aspects of the cloud, including best-effort nature, performance guarantees, and capacity right-sizing, which have not been discussed before.

7. CONCLUDING REMARKS

In this paper, we presented a systematic pricing study for cloud resources. We developed a preliminary framework to model the essential aspects of cloud computing. We then conducted pricing analyses on several unique issues of cloud systems, including resource throttling, performance guarantees, and capacity right-sizing. The results revealed interesting insights to better understand some common industry practices and research directions pertaining to the cloud. As future work, we plan to extend our framework to consider the pricing problem with resource over-provisioning and multiple resources.

8. REFERENCES

[1] `http://aws.amazon.com/ec2/faqs/#What_is_an_EC2_Compute_Unit_and_why_did_you_introduce_it`.

[2] `http://wiki.xensource.com/xenwiki/CreditScheduler`.

[3] `http://gigaom.com/cloud/what-google-compute-engine-means-for-cloud-computing/`.

[4] V. Abhishek, I. A. Kash, and P. Key. Fixed and market pricing for cloud services. In *Proc. NetEcon*, 2012.

[5] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron. Towards predictable datacenter networks. In *Proc. ACM SIGCOMM*, 2011.

[6] L. A. Barroso and U. Hölzle. The case for energy-proportional computing. *Computer*, 40(12):33–37, December 2007.

[7] T. Basar and R. Srikant. Revenue-maximizing pricing and capacity expansion in a many-users regime. In *Proc. IEEE INFOCOM*, 2002.

[8] A. Gandhi, M. Harchol-Balter, R. Das, and C. Lefurgy. Optimal power allocation in server farms. In *Proc. ACM Sigmetrics*, 2009.

[9] A. Ghodsi, V. Sekar, M. Zaharia, and I. Stoica. Multi-resource fair queueing for packet processing. In *Proc. ACM SIGCOMM*, 2012.

[10] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica. Dominant resource fairness: Fair allocation of multiple resource types. In *Proc. USENIX NSDI*, 2011.

[11] Google Cluster Data. `http://code.google.com/p/googleclusterdata/wiki/ClusterData2011_1`.

[12] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel. The Cost of a Cloud: Research Problems in Data

Center Networks. *SIGCOMM Comput. Commun. Rev.*, 39(1):68–73, 2009.

[13] 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. ACM CoNEXT*, 2010.

[14] P. Hande, M. Chiang, R. Calderbank, and J. Zhang. Pricing under constraints in access networks: Revenue maximization and congestion management. In *Proc. IEEE INFOCOM*, 2010.

[15] K. R. Jackson, L. Ramakrishnan, K. Muriki, S. Canon, S. Cholia, J. Shalf, H. J. Wasserman, and N. J. Wright. Performance analysis of high performance computing applications on the Amazon Web Services cloud. In *Proc. IEEE CloudCom*, 2010.

[16] C. Joe-Wang, S. Sen, T. Lan, and M. Chiang. Multi-resource allocation: Fairness-efficiency tradeoffs in a unifying framework. In *Proc. IEEE INFOCOM*, 2012.

[17] S. Li, J. Huang, and S. R. Li. Revenue maximization for communication networks with usage-based pricing. In *Proc. IEEE Globecom*, 2009.

[18] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska. Dynamic right-sizing for power-proportional data centers. In *Proc. IEEE INFOCOM*, 2011.

[19] A. Mas-Colell, M. D. Whinston, and J. R. Green. *Microeconomic Theory*. Oxford University Press, 1995.

[20] J. Mo and J. Walrand. Fair end-to-end window-based congestion control. *IEEE/ACM Trans. Netw.*, 8(5):556–567, October 2000.

[21] D. Niu, C. Feng, and B. Li. Pricing cloud bandwidth reservations under demand uncertainty. In *Proc. ACM Sigmetrics*, 2012.

[22] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer, and D. Epema. A performance analysis of EC2 cloud computing services for scientific computing. In *Cloud Computing*, volume 34, pages 115–131. Springer, 2010.

[23] J. S. Otto, R. Stanojevic, and N. Laoutaris. Temporal rate limiting: Cloud elasticity at a flat fee. In *Proc. NetEcon*, 2012.

[24] J. Schad, J. Dittrich, and J.-A. Quiané-Ruiz. Runtime measurements in the cloud: Observing, analyzing, and reducing variance. In *Proc. VLDB*, 2010.

[25] S. Shakkottai, R. Srikant, A. Ozdaglar, and D. Acemoglu. The price of simplicity. *IEEE J. Sel. Areas Commun.*, 26(7):1269–1276, September 2008.

[26] Y. Shang, D. Li, and M. Xu. Energy-aware routing in data center network. In *Proc. ACM SIGCOMM Workshop on Green Networking*, 2010.

[27] A. Shieh, S. Kandula, A. Greenberg, C. Kim, and B. Saha. Sharing the data center network. In *Proc. USENIX NSDI*, 2011.

[28] The ANL Intrepid Log. `http://www.cs.huji.ac.il/labs/parallel/workload/l_anl_int/index.html`.

[29] The RICC Log. `http://www.cs.huji.ac.il/labs/parallel/workload/l_ricc/index.html`.

[30] V. Valancius, C. Lumezanu, N. Feamster, R. Johari, and V. V. Vazirani. How many tiers? Pricing in the Internet transit market. In *Proc. ACM SIGCOMM*, 2011.

[31] E. Walker. Benchmarking Amazon EC2 for high-performance scientific computing. *USENIX Login*, 33(5), October 2008.

[32] G. Wang and T. Ng. The impact of virtualization on network performance of Amazon EC2 data center. In *Proc. IEEE INFOCOM*, 2010.

[33] H. Wang, Q. Jing, R. Chen, B. He, Z. Qian, and L. Zhou. Distributed systems meet economics: Pricing in the cloud. In *Proc. HotCloud'10: 2nd USENIX Conf. Hot Topics in Cloud Computing*, 2010.

[34] C. Yoo. Network neutrality and the economics of congestion. *Georgetown Law Journal*, 94, June 2006.

APPENDIX

A. PROOF OF THEOREM 1

To solve PD_OPT, first note that $S_v(p) > 0$ for all v and p, and can be ignored. Since it is a convex optimization problem, by using Langrange multiplier technique, we can get the first-order necessary and sufficient condition with respect to p_v:

$$f(v)\big(v^{1/\alpha}(1-1/\alpha)p_v^{-1/\alpha} + \lambda v^{1/\alpha}(1/\alpha)p_v^{-1/\alpha-1}\big) = 0$$
$$\Rightarrow p_v = \frac{\lambda}{1-\alpha}.$$

λ is the Lagrange multiplier corresponding to the capacity constraint. We observe that p_v does not depend on the utility level of individual tenant v, and thus the optimal p_v is uniform across all tenants. PD_OPT and Basic_OPT lead to the same solution, and it suffices to consider Basic_OPT with uniform pricing.

To solve λ, note that the capacity constraint must hold with equality, since the revenue function $R_v(p)$ is strictly decreasing with p. Thus plugging $p = \lambda/(1-\alpha)$ into (8), we can obtain the optimal λ^*:

$$\int_{v_0}^{v_1} \left(\frac{v(1-\alpha)}{\lambda^*}\right)^{1/\alpha} f(v)\mathrm{d}v = C$$

The optimal price p^* then follows, and Theorem 1 is immediate.

B. PROOF OF THEOREM 3 AND LEMMA 1

Notice from (15) that revenue increases with q. Thus for the individual rationality constraint to hold for all tenants, $S_{v_0}(p_s^*, q^*) = 0$, i.e. $q^* = \frac{\alpha v_0^{1/\alpha}}{1-\alpha} p^{1-1/\alpha}$. Substitute into (15), we can see revenue increases when p_s^* decreases. Thus the optimal price can be found when the capacity constraint in SLA_OPT is satisfied at equality. This argument is essentially the same as the Lagrange multiplier argument above. The optimal surplus S_s^* and revenue R_s^* can then be readily derived from (14) and (15), respectively.

$S_s^* > S_t^* = 0$ is obvious. To show $R_s^* < R_t^*$, from Theorem 3 and 2,

$$\frac{R_s^*}{R^*} = 1 + \frac{\alpha v_0^{1/\alpha}}{(1-\alpha)B} < 1 + \frac{\alpha}{1-\alpha} = \frac{R_t^*}{R^*}$$

for $B = \int_{v_0}^{v_1} v^{1/\alpha} f(v)\mathrm{d}v > v_0^{1/\alpha}$.

C. PROOF OF THEOREM 4

Using the Lagrange multiplier technique, the KKT conditions of Cost_OPT can be written as:

$$\alpha - 1 + \frac{e_1 + \lambda}{p^*} = 0,$$
$$\lambda(C - B(p^*)^{-1/\alpha}) = 0,$$
$$C - B(p^*)^{-1/\alpha} \geq 0.$$

$\lambda \geq 0$ is the Lagrange multiplier to the capacity constraint. Directly setting the first-order derivative of the profit $R_v(p) - E_v(p)$ to zero we otain

$$p_c = \frac{e_1}{1-\alpha}.$$

At p_c, the total demand D_c equals $B(\frac{e_1}{1-\alpha})^{-1/\alpha}$. If $D_c < C$, i.e. $e_1 > (1-\alpha)B^\alpha C^{-\alpha}$, p_c constitutes a solution that satisfies all the KKT conditions with $\lambda = 0$.

If $D_c \geq C$, i.e. $e_1 \leq (1-\alpha)B^\alpha C^{-\alpha}$, p_c is not a feasible solution. Thus the optimal price must be larger than p_c. The profit function $R_v(p) - E_v(p)$ can be shown to be concave in p, and thus decreases with p in the interval $[p_c, +\infty)$. Therefore, $p^* = B^\alpha C^{-\alpha}$, i.e. the optimal price is the smallest feasible price at which demand equals capacity.