1-2013

The pricing model of Cloud computing services

Jianhui HUANG
Singapore Management University, jhhuang.2009@smu.edu.sg

Dan MA
Singapore Management University, madan@smu.edu.sg

The Pricing Model of Cloud Computing Services

Dan Ma
School of Information Systems
Singapore Management University
Singapore
madan@smu.edu.sg

Jianhui Huang
School of Information Systems
Singapore Management University
Singapore
Jhhuang.2009@smu.edu.sg

ABSTRACT

Cloud computing service providers offer computing resource as a utility and software as a service over network. Many believe that Cloud computing is making an industry-wise paradigm shift for IT use. Besides its technique issues, the business feature of Cloud computing attracts our interests. Specifically the practice of Amazon EC2 introduces an interesting pricing scheme. Amazon provides users with virtual computing instances as a combination of interruptible service (i.e., spot instance) and uninterruptible service (i.e., on-demand and reserved instance). Spot instance is charged at a per use price which is dynamically changing over time; users of spot instance face the risk of service termination on the provider's side. In this paper, we build a multiple-stage game to study users' best job submission strategy under such a mixed pricing scheme, and to analyze the potential benefits and influence of such pricing scheme. We identify user-segments that will take different job submission strategies, and show that users should reserve resource for future high-value job arrivals. We also conduct numerical investigations to demonstrate how the outcome and best strategy vary according to the external market conditions.

Categories and Subject Descriptors
H.1.0 [**Models and Principles**]: General

General Terms
Economics

Keywords
Cloud Computing, Pricing model, Multi-stage Game

1. INTRODUCTION

Cloud Computing has attracted great interests from both the industry and academy in the past one decade. It offers a paradigm shift towards computing served as a utility and software accessed as a service over the network. Importantly, Cloud Computing should not be only considered as a technical innovation itself. Sharing computer resources in Grid computing, virtualization and on demand services such as Software-as-a-Service (SaaS) has been there before Cloud Computing appeared. The real revolution of Cloud is the combination of those different IT aspects into a new business model that shifts the traditional way of IT provisioning.

Practitioners, both firms who are in need of IT resources and providers who offer IT capacity are trying to understand this new business model and figure out how to benefit from it. Although usage-based pricing model is adopted by most providers, there are different versions of it in practice. For example, Salesforce.com[1], probably the most successful Software-as-a-Service (SaaS) provider so far, charges a monthly subscription fee to its users for accessing its online CRM software applications. Amazon Elastic Computing Cloud (EC2)[2], the leading Infrastructure-as-a-Service (IaaS) provider, charges users by hour. Cloud Sigma[3], an emerging IaaS provider from Switzerland, also charges customers by hour, but its billing segment is as short as 5 minutes. The focus of our study is the pricing model for Cloud service providers. Specifically, we identify the possibility of simultaneously using multiple pricing methods for better resource utilization, and study how it could potentially benefit both the provider and/or users.

Amazon EC2 is a well-established cloud computing service. There are three types of EC2 instances. On-demand computing instance was first introduced in 2006. After that, reserved instance was introduced in early 2009, and spot instance in late 2009 [1]. A spot instance is charged at a "spot price." To use a spot instance, a user must submit a bid price; Amazon collects all the requests and bids from users, and decide the "spot price". A spot price therefore is set based on the relationship of supply and demand dynamically over time. The price history over the past 90 days is publically available on the Amazon website, which helps a new user to estimate the cost of spot instance and to submit an appropriate bid. A very interesting and unique feature of spot instance, other than its dynamic price, is its service interrupt-ability. The right of service termination is in the hands of the provider. For example, at time t, if a user's bid price exceeds the spot price, the user's requested spot instance would be launched. However, at time t+1, if the spot price increases while that instance is still running, Amazon will terminate it. This instance fails to execute. It is no doubt that users of spot instance face uncertainty risk of service-being-terminated. The "expected" quality of spot instance therefore is lower than that of on-demand or reserved instance, since the latter two instances both have

1 It charges US$65/per user/month for it professional complete CRM applications for any size team. Please refer to http://www.salesforce.com/ap/crm/editions-pricing.jsp for more tiered pricing schemes.

2 Amazon EC2 charges $0.08 per hour for standard small instance in us-east-1 region. More price information is available at http://aws.amazon.com/ec2/pricing/.

3 CloudSigma charges $0.0106 per hour for CPU power and updates price every 5 minutes. Please refer to http://www.cloudsigma.com/en/pricing/price-schedules for more information.

guaranteed services without interruption after paying the respective price. Both on-demand and reserved instances give full control to customers and are of the same quality. Reserved instances are cheaper because it requires users pay a certain amount of upfront fee to reserve the capacity [2]. It requires customers to anticipate and schedule their task executions given their past experience with their own IT demand. Spot instance is the cheapest among the three. Risk seeking users might run all their jobs on spot instance to pursue minimum cost on computing service. However they have to take the risk of losing services unexpectedly.

The introduction of spot instance is consistent with some previous studies. Bhargava and Choudhary [4] stated that it's optimal for a software vendor to offer lower-quality product when the variable cost is low. Customers' perceived quality of spot instance is lower because it could be terminated unwillingly by the provider. In addition, Etzion [5] showed that an optimal design of dual selling channel, auction and fix price selling, could outperform pure selling in most circumstances.

Like traditional IT service providers, Cloud computing service providers must strategically manage the peak load capacity and overall utilization level of their infrastructure. The infrastructure of a Cloud service provider typically is not fully utilized in most of the time [7]. In addition, due to the advance in virtualization technology and emergence of Internet based delivery model, there is no need to physically transfer facilities from one place to another in order to fulfill requirements of globally distributed customers. Thus the cost of reallocating resources is negligible for Cloud computing services. Cloud vendor is able to and should pull the IT demand together from the whole market so that they could better manage resources and handle demand volatility. The use of spot instance illustrates that Amazon's practice of encouraging customers to use their spare computing resource that is not occupied by reserved and on-demand instances.

In the real world, thousands of instances are simultaneously running in the Cloud. A naive yet reasonable way to group the various instances is to divide them into two types: important high value jobs and low value jobs. To serious business users, high value tasks are those keep their core business functionality running with a reasonable high performance. Low value tasks are those help to enhance user's experience in terms of response time, resolution level, and so on. A profit maximizing business user will find Amazon EC2 especially attractive as it provides two types of instances that fit the two types of jobs. Given the very low cost of spot instance, a risk seeking customer might put its jobs on spot to enjoy the great cost reduction but bears the risk of service interruption. There are some real world examples that users submit almost all its jobs to spot instance. SEOMOZ, a Cloud-based SEO software vendor, runs most of their services on Amazon spot instance before September 2011. A spot price spike happened during September 2011. Amazon terminated all their spot instances. It took SEOMOZ two weeks to fully recover the services [10]. After the accident SEOMOZ changed to a mix strategy that uses both on-demand instance and spot instance.

From the accident of SEOMOZ, we see an effort of current Cloud computing customers to adapt to the offered service category and pricing model. The concept of spot instance is fresh in the context of computing service. In the past, the privilege of job termination was always in a customer's hand. In spot instance, it is shifted to the service provider's hand. Although spot instance customers are facing a lower unit price, the loss from service termination could be much higher than the cost reduction, as indicated in the SEOMOZ example. Cost reduction cannot always efficiently leverage the risk. In such cases customer who rely their business on spot service should employ a mixture usage of quality guaranteed service and spot service to lower the average risk to an acceptable level.

There are studies that proposed solutions for utilizing spot service efficiently through strategic bidding [14]; and studies that introduced option mechanism to the spot service market [8]. In this study we take a different research angle. We are interested in the marketplace with one Cloud service provider and many customers in need of IT resources. Customers have anticipated IT demand and they use the mixture of reserved service and spot service to leverage risk. Knowing the strategic behavior of customers, the Cloud service provider tries to set prices, for both reserve instance price (which is fixed) and spot instance price (which is dynamically adjusted over time), in order to (1) maximize profit and (2) manage its IT resources effectively.

Many previous works have analyzed and compared the selling model with a fixed fee and renting model with a pay-by-usage pricing [6;11;12]. Their major findings indicated that the fixed fee scheme could add value to the usage based pricing or sometimes outperform the usage based pricing in a competitive setting. Similar to these works, we also look at a monopolistic service provider who adopts two pricing schemes simultaneously. But different from these pervious works, we not only consider different pricing schemes, but also differentiated services. In addition, in our setting, customers face jobs with different nature.

This study contributes to the Cloud computing literature in that it analyzes the mixed adoption model in the Cloud industry. It helps shed lights on resource / capacity management for Cloud service providers. It is also the first work, to our best knowledge, to study interruptible service in a business context.

2. THE MODEL

In the market, there are many risk neutral customers and a monopoly Cloud service provider. The provider offers two types of services, reserved service and spot service, simultaneously. Reserved service is charged at a fixed fee with an upper limit in usage amount, and spot service is charged by a unit price that is dynamically adjusted over time.

We model it as a multi-stage game. There are k stages in the timeline. Customers must make a purchase decision of reserved service at the first stage. When a job arrives, customer will submit to the Cloud provider, either as a reserved instance, if this customer purchased reserved service at the first stage, or as a spot instance. For analytical convenience, we assume that customers only submit jobs at the beginning of each stage. The job arrival for customer i's during one stage follows a Poisson distribution, arrival rate at λ_i We assume for one customer faces the same arrival rate in all stages and the job arrival process in all stages are i.i.d. But different customers will face different λ, reflecting the nature difference in customers' IT demand. There are two types of jobs, high value V_H and low value V_L. The proportion of high value jobs is a. This proportion keeps invariant for all customers. At the beginning of each stage, a customer must choose to submit its job arrivals to reserved service or spot service. Note that the price of using reserved service has been fixed at the first stage but there is an upper limit of amount a customer could use, while the price of spot instance is varying each period and the customer might be forced to terminate the job if the spot price rises in the next stage.

Reserved service is charged at a fixed price T with N units of instance hours. Spot price fluctuates randomly with a mean of p_s which are known by all customers. We make such an assumption based on the following observations. In the website of Amazon EC2 service, the past 90-days spot price history is offered as the public information. Knowledge of how the spot prices change over time, such as its mean, volatility level, and varying pattern is available to potential customers. Moreover, Ben-Yehuda, et. al. [3] pointed out that spot price is randomly set within certain boundaries. Richard, et. al. [9] tested the time interval between two successive price change events in an 80 days price trace of standard small Linux spot instance, and his result showed price actually is changing randomly.

Furthermore, we assume the probability of an interruption (for spot instance) is a random variable with mean p_t. Jobs that are submitted to spot service and finally interrupted will not be resubmitted again. So there would be loss of utility for customers in the case of service interruption. We assume, at stage t, spot price is p_{st} and the probability of service interruption is p_{tt}. Both are exogenous given and known to customers at the beginning of stage t. But customers don't know the future spot price (at stage t+1), nor the future service interruption probability.

Table 1. Variable definition

Variable	Definition
N_{it}	Reserved service instance hours left at the beginning of stage t
λ_i	Job arrival rate of customer i in each stage
R_{it}	Number of jobs submitted to reserved service in stage t
S_{it}	Number of jobs submitted to spot service in stage t
SS_{it}	Number of jobs that are submitted to spot service in period t and successfully finish without service interruption

The Cloud provider's decision variables are T and N, the reserved instance parameters. They must be determined at the first stage and keep invariant over time. For customers, at stage one, they have to decide whether or not to purchase the reserved service given (T, N). At the consecutive stage i (i>1), they have to decide how many jobs to submit to reserved instance if they purchased the reserved service at the first stage, and how many to submit to the spot instance.

3. ANALYSIS

We assume the value of high-value job, V_H, is high enough that in case of $p_s < \frac{T}{N}$, the cost reduction in using spot service cannot offset a customer's utility loss when a high-value job, submitted to spot service, is interrupted. That is to say $p_s + V_H p_t > \frac{T}{N}$. Thus the more V_H jobs a user potentially has, the more demanding the user is for reserved service. Denote $V_0 = aV_H + (1-a)V_L$, which measure the average job value, and $N(p_s + V_0 p_t) > T$. This assumption indicates customers with N or more than N jobs will be better off using reserved service compared to using spot service. The reason is that if a customer has N jobs, he can execute all jobs on reserved service and get net utility $NV_0 - T$. But the net utility becomes $NV_0(1-p_t) - Np_s = NV_0 - N(p_s + V_0 p_t)$ if he uses spot service instead, which is lower. The utility difference is $N(p_s + V_0 p_t) - T$. When a customer has more than N jobs, he should submit all V_H jobs and some V_L jobs to reserved service, in which case the average utility of jobs executed on reserved service is higher than V_0. So it suggests that customers still get higher net utility using reserved service, and the utility difference could be even larger. Therefore the conclusion that customers with N or more than N jobs is better off using reserved service than using spot service follows.

Customers vary in demands in terms of the number of jobs. So they face different job arrival rates. In what follows, we identify customers who will purchase reserved service. Customer i expects to have $k\lambda_i$ jobs in total the whole process of k stages. If he has purchased the reserved service, his expected utility is calculated as follows:

$$E(U_i) = E(U_i | 0 < n_i < N) + E\left(U_i \middle| N \le n_i \le \frac{N}{a}\right) + E(U_i | n_i > N)$$

Extending this expression, we get:

$$E(U_i) = \int_0^N V_0 t f(t)dt + \int_N^{N/a} \left((V_H - V_L)at + NV_L\right)f(t)dt + \int_{N/a}^{\infty} V_H t f(t)dt$$

When a Poisson distribution has a relative big arrival rate λ, we can use a Normal distribution with mean λ and variance λ as approximation of the Poisson distribution. When λ is greater than 20, it is sufficient for such approximation. Thus we can use the below formula to calculate $E(U_i)$.

$$f(t) = \frac{1}{\sqrt{2\pi\lambda}} e^{-\frac{(t-\lambda)^2}{2\lambda}}$$

It is easy to observe that $E(U_i)$ increases with λ. The customer who is indifferent regarding purchase reserved service or not has a job arrival rate λ^0 that makes $E(U_i) - T = 0$.

The more jobs a customer has, the more likely he is going to purchase the reserved service. Intuitively, when the total number of jobs is close or exceeds the reserve service upper limits N, the user will submit more V_H jobs running on reserved instance. Thus the average utility he gets from utilizing one unit of reserved service increases.

We next identify customers that will use a mixture of reserved service and spot service. The "first" customer in this group should be the one whose jobs all run on reserved service and expected utility is maximized given the default N reserved instance hours.

In order to find this specific user, indicated by the arrival rate λ^*, we have to solve the maximization problem as below.

$$N^* = argmax_{N^*}\left(\int_0^{N^*} (V_0 t - p^* N^*) f_{k\lambda}(t)dt + \int_{N^*}^{\infty} (V_0 - p^*)\, N^* f_{k\lambda}(t)dt\right)$$

Here $p^* = \frac{T}{N}$ stands for the virtual unit price of reserved instance. Solve the maximization problem we can get N^* for any specific customer with demand represented by job arrive rate $k\lambda$ over k stages.

$$N^* = F_{k\lambda}^{-1}(\frac{V_0 - p^*}{V_0})$$

For every λ there is an N^*, and N^* increases in λ. λ^* is the one that makes N^*=N. Customers whose λ are greater than λ^* will use both reserved service and spot service. They can actually execute a greater ratio of V_H jobs on reserved instance and get higher utility than V_0 per instance hour. Customers who purchased reserved service and has arrival rate $\lambda<\lambda^*$ will execute all jobs on reserved service.

We assume the N reserved instance hours can cover all V_H jobs for any individual customer. That is to say, the customer whose λ is the max is able to execute all its V_H jobs on reserved service, i.e. $a\lambda \leq N$. However in reality customer can have more than $\frac{N}{a}$ jobs. In such a case, a customer will divide its jobs into multiple groups, each group satisfying the condition $a\lambda \leq N$. Then we can derive the largest λ that represents the customer using reserved service to execute V_H jobs only. We still can apply the utility maximization rule and derive the max arrival rate denoted as $\bar{\lambda}$. It should satisfy the condition:

$$N = F_{k\bar{\lambda}}^{-1}(\frac{V_H - p^*}{V_H})$$

Here we use a numerical example to illustrate how to calculate λ^0 and λ^*. Consider there is only one stage, k=1. Model parameters are as shown in Table 2.

Table 2. Parameter values of numerical illustration (for the calculation of λ^0 *and* λ^*)

Variable	T	N	V_H	V_L
Value	69	200	0.8	0.4

These values are set subject to the constraint:

$$aV_H + (1 - a)V_L > \frac{T}{N}$$

When we change the proportion of high-value jobs in the total job arrivals, a, the results are shown in Table 3.

Table 3. Results of numerical illustration of λ^0 *and* λ^*

A	0.2	0.25	0.3	0.32
λ^0	144	139	133	131
λ^*	208	207	206	206
$\bar{\lambda}$	990	792	660	618
$\sigma_1(\lambda^*- \lambda^0)$	64	68	73	75
$\sigma_2(\bar{\lambda} - \lambda^*)$	782	585	454	412

The numerical examples show that when a increases, λ^0 decreases, λ^* slightly decreases, and $\bar{\lambda}$ decreases a lot. The interval between λ^0 and λ^* increases while the interval between λ^* and $\bar{\lambda}$ decreases. The larger a is, the more customers will purchase the reserved service, and among them the more customers will use a combination of reserved service and spot service.

Lemma 1 identifies some behavior of a customer after he purchased reserved instance.

Lemma 1. A customer, after purchasing reserved instance, will submit V_H jobs to reserved instance so long as there is resource remain unused.

[Proof] If a high value job arrives and the customer doesn't submit it to reserved instance, there could be three possible outcomes.

(1). This customer runs out of reserved resource at some stage afterwards.

(2). At the last kth stage, remained reserved resources are just enough to satisfy all V_H jobs arrived in that stage.

(3). At the last kth stage, after submitting all V_H jobs to reserved service, there is still some reserved resource left.

In cases (1) and (2) there is no extra cost. In case (3) this customer bears some efficiency loss. In this case the best he can do is to submit V_L jobs to reserved instances. While in this case, if he submits the V_H task to reserved service at the current stage the overall utility will be higher. Because submitting V_H to spot instances will cause higher loss if the instance is interrupted. Therefore in this case customer will be better off by submitting current V_H job to reserved service.

To summary, submitting current high value task to reserved service is always no worse than not submitting. Therefore the customer should submit high value tasks to reserved service as long as there are available resource. **Q.E.D.**

Thus in each stage, a customer submits all V_H jobs, possibly some V_L jobs too, to reserved service instantly. Never submitting V_L jobs to reserved service is not optimal. The reason is that in the last kth stage, after submitting all jobs to reserved service, there is positive probability that there is still some reserved resource unused. The waste can be avoided if in current stage customer submits sufficient jobs, including V_L jobs, to reserved service. But customer should do this with care, as a rash decision could cause running out of reserved resource before submitting all V_H jobs in the last stage. Let's denote the number of low value jobs submitted to reserved service by x.

Lemma 2. When $\frac{V_0-p^*}{V_0} < 0.5$, customers tend to submit low-value jobs to reserved service only if they are in later stage of the game.

Prove: If a customer submits a V_L job to reserved service and later on he finds there is no enough resource for V_H jobs, the potential loss of utility is p_s+(1-p_f)V_H. When the customer doesn't submit a V_L job to reserved service and in the end there is resources unused, the potential loss in utility is p_s+(1-p_f)V_L. Since V_H>V_L, among the two cases delaying the submission of low value jobs to reserved service is preferred because it causes less potential loss in utility.

However the discussion above assumes that the two probabilities are equal, i.e. the possibility that reserved service is going to run out, and the possibility that reserved service is going to be wasted.

In order to maximize his utility, a customer should keep $F^{-1}(\frac{V_0-p^*}{V_0})$ units of reserved instance hours for future use. [4] When $\frac{V_0-p^*}{V_0} < 0.5$, the probability of resources shortage $1 - \frac{V_0-p^*}{V_0}$

[4] Recall that F(x) is the cumulative density function of a normally distributed random variable with both mean and variance equal to the customer's aggregated job arrival rate in future stages.

is larger than that of over-provision $\frac{V_0-p^*}{V_0}$. Therefore, in this situation, instead of submitting all jobs to reserved service, they will use reserved service to execute all high value jobs and delay submitting low value jobs to reserved service.

Q.E.D.

At any stage t, customer i got n_{it} jobs. Denoting the number of V_H jobs by H_{it} and the number of V_L jobs by L_{it}. The customer will submit min(H_{it}, N_{it}) high value jobs to reserved service. Besides, he will submit x_{it} low value jobs to reserved service, $0 \leq x_{it} \leq min(max(0, N_{it}-H_{it}), L_{it})$. In total, he submits R_{it}=min(H_{it}, N_{it})+x_{it} jobs to reserved instance. In addition, S_{it} jobs are submitted to spot service. Note that R_{it} might be smaller than H_{it} and it could include both V_H and V_L jobs.

We can use a list <N_{it}, n_{it}, R_{it}, S_{it}, SS_{it}> to represent customer i's status and decision in stage t. They satisfy:

$$R_{it}+S_{it}=n_{it}$$

$$N_{it+1}=N_{it}-R_{it}=N_{it}-an_{it}-x_{it}\ (SS_{it} \leq S_{it})$$

$$x_{it} \in (0, \min(\max(0, N_{it}-H_{it}), L_{it}))$$

3.1 The 2-stage case

In a 2-stage game (k=2), stage 1 is the initial stage in which customer can purchase reserved service, and stage 2 is the final stage in which customers can decide what type of jobs go to what type of service. We analyze such a game backwards.

Assuming the customer i has purchased reserved service at stage 1. In stage 2 there are 3 possible conditions with respect to the relationship between n_{i2} and N_{i2}, and in each condition, the customer's best choice on x_{i2} is determined as given:

$$x_{i2} = \begin{cases} (1-a)n_{i2}, & 0 < n_{i2} < N_{i2} \\ N_{i2} - an_{i2}, & N_{i2} \leq n_{i2} \leq \dfrac{N_{i2}}{a} \\ 0, & n_{i2} > \dfrac{N_{i2}}{a} \end{cases}$$

Then expected x_{i2} is:

$$E(x_{i2}) = \int_{0}^{N_{i2}} (1-a)tf(t)dt + \int_{N_{i2}}^{N_{i2}/a} (N_{i2}-at)f(t)dt$$

Here $N_{i2}=N_{i1}-H_{i1}-x_{i1}$. The customer's best choice of x_{i2} is a function of x_{i1}. It is not surprising that customers facing more high value jobs (a bigger a) tend to use less reserved resource to execute low value jobs.

Given the distribution of n_{i2} and utility structure of customer i, he can decide an optimal number of reserved hours for V_H jobs. Since stage 2 is the final stage, job arrivals at stage 2 could be considered equivalent to the dynamic demand modeled in a single selling season in the newsvendor model [13]. Therefore the optimal number of reserved instance for future V_H jobs, which will be kept from executing any type of job, is:

$$\overline{H}_{i2} = aF_{\lambda_i}^{-1}(\frac{V_H - p^*}{V_H})$$

In stage 1, the customer receives n_{i1} job. He has to choose an optimal x_{i1}. When $H_{i1} + \overline{H}_{i2} \geq N$, this customer will set x_{i1}=0. When $H_{i1} + \overline{H}_{i2} < N$, he solves the problem below to get optimal value of x_{i1}.

$$x_{i1} = argmin_{x_{i1}}\left(N - \left(H_{i1} + \overline{H}_{i2} + x_{i1} + E(x_{i2})\right)\right)$$

Subject to:

$$H_{i1} + \overline{H}_{i2} + x_{i1} + E(x_{i2}) \leq N$$

Note $H_{i1}=an_{i1}$. The problem is equivalent to maximize $x_{i1} + E(x_{i2})$ subject to the upper bound of $N - H_{i1} - \overline{H}_{i2}$. Although $E(x_{i2})$ decreases in x_{i1}, it's not clear how the combination moves with x_{i1}. We use a numerical example to illustrate how the best x_{i1} looks like. Parameter setting is as shown in Table 4.

Table 4. Parameters of numerical illustration of x_{i1}

Variable	T	N	V_H	λ	n_1
Value	69	200	0.8	160	155

The parameters are set subject to the two constraints: 1) $\frac{T}{N} < V_H$ and 2) $N \geq 2a\lambda$; . We vary the value of a. Results are shown in following figures (Figure 1a ~ 1d).

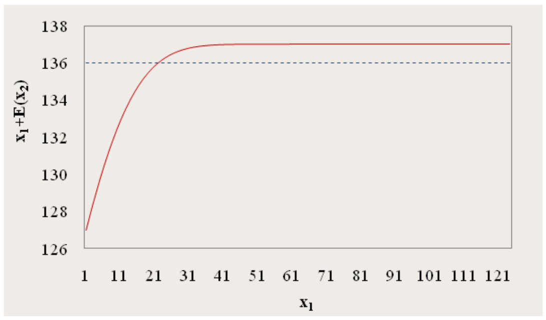

Figure 1a. a=0.2

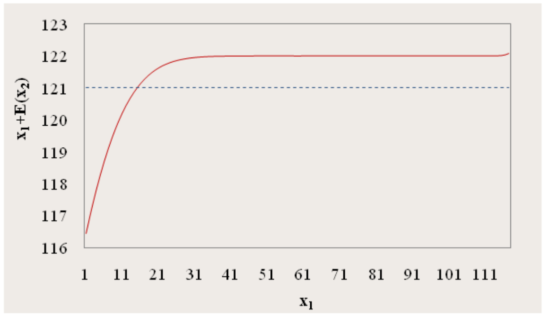

Figure 1b. a=0.25

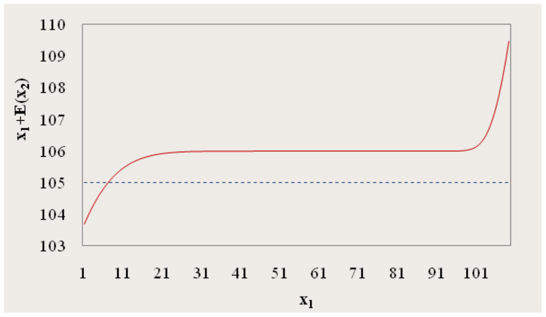

Figure 1c. a=0.3

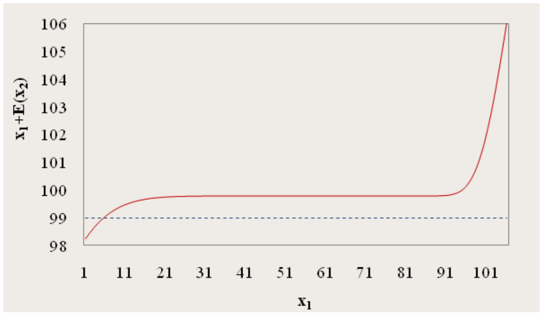

Figure 1d. a=0.32

The horizontal dotted line in these figures represents $N - H_1 - \bar{H}_2$. As a increases, x_{i1} decreases. The customer has to "hold" more reserved resources for high value jobs in stage 1 when he is planning the stage 2 usage. The numerical examples show us that x_{i1} is much smaller than x_{i2}. The behavior is consistent with our observations in lemma 2 that as the proportion of V_H jobs increases, customers are more likely to submit V_L jobs to reserved service only in stage 2.

Figure 1c and 1d demonstrate a different shape of curve, compared to Figures 1a and 1b. We see the curve of $x_{i1}+E(x_{i2})$ has fast raising in both ends, but a flat line in the middle. In addition, we observe through a large scale of numerical examples[5] that when V_H is getting close to V_L, the middle flat line moves upwards and two rising ends stay invariant. In addition, as V_H is getting closer to V_L, the optimal value of x_{i1} will increase as well. It implies that when the difference between high value jobs and low value jobs diminishes, customers are less likely to hold "space" in reserve service for future high-value job arrivals. Rather, they tend to submit current demand (both high and low value jobs) to reserve instances.

3.2 The k-stage case

The analysis for 2-stage game shows that customers tend to hold a certain amount of reserved resource in earlier stages to make sure there are sufficient reserved instance hours for the future V_H jobs.

Now consider the k-stage case. Customers have the similar concern as that in the 2-stage game. Customers have to consider to issues. First, how many reserved instance hours to be kept for future V_H jobs, and second, how many V_L jobs to submit to the reserved instance at the current stage.

The logic behind the considerations is that if current stock of reserved resource significantly exceeds optimal the stock level for future V_H jobs, some of the reserved resource will likely be wasted if a customer takes a too conservative strategy in submitting V_L jobs to reserved service. On the other hand, if a customer is too aggressive in submitting V_L jobs to reserved service, he faces the risk of running out of reserved resource before the last batch of V_H jobs arrive.

Each customer's job arrival process is independent and identical among k stages. In this case, multiple i.i.d Poisson arrival processes can be combined as a single Poisson arrival. Thus for customer i in stage t, the number of its total future jobs, in the rest of k-t stages, denoted by $\bar{n}_{it+1}$, follows a Poisson distribution with the arrival rate (k-t)λi.

[5] Due to the length limit, results from these large set of numerical examples are not provided here, but they are available upon request.

Similar to the 2-stage case, at any stage t, the customer i can foresee three conditions regarding $\bar{n}_{it+1}$.

(1). $0 < \bar{n}_{it+1} < N_{it+1}$

(2). $N_{it+1} \le \bar{n}_{it+1} \le \frac{N_{it+1}}{a}$

(3). $\bar{n}_{it+1} > \frac{N_{it+1}}{a}$

Here $N_{it+1}=N_{it}-R_{it}=N_{it}-H_{it}-x_{it}$.

We can assume a virtual stage $\bar{t}$ as the combination of the rest k-t stages. We simulated 5 stages Poisson arrival processes with arrival rate 100 and one single stage Poisson arrival process with arrival rate 500. Each has 500 traces. A paired t-test shows there is no difference between the two series generated by these methods. Therefore at stage t, a customer faces similar problem as that in stage 1 of the 2-stage game. However it has to keep different amount of reserved instance hours for future high value jobs based on how many stages there are in the virtual stage. That is to say, in a single stage, the best decision is to keep $\bar{H}$ reserved instance hours for V_H jobs that are to be realized, and $\bar{H} = aF^{-1}(\frac{V_H-p^*}{V_H})$. In this virtual stage $\bar{t}$, the customer will need to keep $\bar{H}_{\bar{t}} = a(k-t)F^{-1}(\frac{V_H-p^*}{V_H})$ reserved instance hours.

So far we can summarize a customer's best decision rule at any stage t:

(1) Submit H_t high value jobs to reserved service.

(2) Calculate $\bar{H}_{\bar{t}}$ using the above formula.

(3) If $\bar{H}_{\bar{t}} \ge N_t - H_t$, the customer should not submit any V_L job to reserved service. Otherwise he will choose the optimal x_t that solves the problem:

$$x_t = argmin_{x_t}\left(N_t - \left(H_t + \bar{H}_{\bar{t}} + x_t + E(x_{\bar{t}})\right)\right)$$

Subject to:

$$H_t + \bar{H}_{\bar{t}} + x_t + E(x_{\bar{t}}) \le N_t$$

And here

$$N_{\bar{t}} = N_t - H_t - x_t$$

$$E(x_{\bar{t}}) = \int_0^{N_{\bar{t}}} (1-a)yf_{\bar{t}}(y)dy + \int_{N_{\bar{t}}}^{N_{\bar{t}}/a} (N_{\bar{t}} - ay)f_{\bar{t}}(y)dy$$

$$f_{\bar{t}}(y) = \frac{1}{\sqrt{2\pi(k-t)\lambda}} e^{-\frac{(y-(k-t)\lambda)^2}{2(k-t)\lambda}}$$

The formulas are similar to those in the 2-stage game, except that the stage 2 is replaced by the our virtual stage $\bar{t}$, which makes the calculations conducted in different scales. The process is repeated until the end of stage k. So we can conclude that a k-stage game is to repeat the 2-stage game (k-1) times.

4. DISCUSSION AND CONCLUSION

We model a multi-stage game with a monopolistic Cloud service provider who offers both reserved service and interruptible spot service. Reserved service is quality-guaranteed, and must be purchased at the beginning by a fixed price. Spot service is interruptible by the provider; and the price for spot service is adjusted dynamically over time. Customers receive two types of job arrivals, high value and low value jobs. They must decide (1) whether to purchase the quality-guaranteed reserve price at the first stage; (2) in each stage, how many jobs to submit to reserve service and how many to submit to spot service.

To analyze this decision-making problem, we first study the equilibrium market segmentation. We define the customer who is indifferent between purchasing and not purchasing reserved service at the first stage. Only those with higher demand than this indifferent customer, in terms of the expected number of job arrivals, will purchase reserved price service. Furthermore, these customers could be categorized into two groups, those who will submit jobs to both reserved service and spot service and those who will only use reserved service. Together with the customers who won't purchase reserved service initially (and hence they only use spot service), we get a three-segment customer population. Numerical results show that the proportion of high value jobs is a major factor influencing the market segmentation. When this proportion increases, i.e., a higher value of a, the segment of customers purchasing reserve service enlarges; for those who use both types of services, they are more likely to submit high value jobs to spot service.

In a 2-stage game setting, we demonstrate how customers should optimally choose stock level of reserved resources and make the optimal decision on how many low value jobs to be executed on reserved instances. In a k-stage case, we make use of virtual stage to represent all future job arrivals and shape customers' decision problem. We prove that customers' behavior is similar to that in the 2-stage case; and the solution could also be derived the same way as that in the 2-stage game.

We plan to extend our work in the following two directions. We first need investigate the decision problem of Cloud provider. So far, we have studied customers' strategic behavior. To us, an equally and even more important research question is that given customers' response, how shall the Cloud provider set the price and quality limit for his reserved instances? How could these decision choices (both p and N in our model) respond to the changes in market conditions, such as customers' job arrival rate, the proportion of high-value job, and/or the job value variation? Secondly, we need use simulations to test our findings for a larger set of parameter values. We expect to discover systematic relationships between the model outcomes and some key parameters.

Our analysis has some limitations. For example, we assume the job arrival process of a customer during all stages is identical and independent, following a Poisson distribution. But in reality the job arrival processes in different stages could be very different and highly related. In addition, in this study, we the spot price p_s and service interruption probability p_t as exogenously given. However, a profit-maximizing Cloud provider in practice should be able to collect user's bid (for spot instance) and set spot price optimally. Such a decision-making of providers is not in our considerations.

5. REFERENCES

[1] Amazon. 2009. Announcing Amazon EC2 Spot Instances. Accessible at http://aws.amazon.com/about-aws/whats-new/2009/12/14/announcing-amazon-ec2-spot-instances/

[2] Amazon. AWS Economics Center. Accessible at http://aws.amazon.com/economics/

[3] Ben-Yehuda, O.A., Ben-Yehuda, M., Schuster, A., and Tsafrir, D. 2011. Deconstructing Amazon EC2 Spot Instance Pricing. In Proceedings of the IEEE Third International Conference on Cloud Computing Technology and Science (Athens, Greece, November 29 - December 1, 2011). CloudCom '11. IEEE Computer Society, Washington, DC, USA, 304-311.

[4] Bhargava, H. K., and Choudhary, V. 2008. Research Note---When Is Versioning Optimal for Information Goods? Manage. Sci. 54, 5 (May. 2008), 1029-1035.

[5] Etzion, H., Pinker, E., and Seidmann, A. 2006. Analyzing the simultaneous use of auctions and posted prices for online selling. Manuf. Serv. Op. 8, 1 (Winter 2006), 68–91.

[6] Fishburn, P.C. , Odlyzko, A.M., and Siders, R.C. 1997. Fixed fee versus unit pricing for information goods: competition, equilibria, and price wars. First Monday. 2, 7 (July 1997). Also to appear in Internet Publishing and Beyond: The Economics of Digital Information and Intellectual Property, D. Hurley, B. Kahin, and H. Varian, eds. MIT Press.

[7] Liu, H. 2011. A Measurement Study of Server Utilization in Public Clouds. In Proceedings of the 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing (Sydney, Australia, December 12-14, 2011). DASC '11. IEEE Computer Society, Washington, DC, USA, 435-442. DOI=10.1109/DASC.2011.87 http://dx.doi.org/10.1109/DASC.2011.87

[8] Rahman, M. R., Lu, Y., and Gupta, I. 2011. Risk aware resource allocation for clouds. Technical Report. University of Illinois at Urbana-Champaign. Available at http://hdl.handle.net/2142/25754

[9] Richard, D.S., Huang, J.H., and Yang, Y.P. 2012. Negotiable SLAs for Cloud Computing Spot Market: a Myth or a Business Opportunity. Working paper.

[10] SEOMOZ. 2011. Crawl Outage - An Update and What We're Doing. Accessible at http://www.seomoz.org/blog/crawl-outage

[11] Sridhar, B., Bhattacharya, S., and Krishnan, V. 2009. Pricing Information Goods: A Strategic Analysis of the Selling and On-demand Pricing Mechanisms. Working paper.

[12] Sundararajan, A. 2004. Nonlinear Pricing of Information Goods. Manage. Sci. 50, 12 (Dec. 2004), 1660-1673.

[13] William, J. S. 2009. Operations Management. 10th edition, McGraw-Hill/Irwin, 581.

[14] Yi, S., Kondo, D., and Andrzejak, A. 2010. Reducing costs of spot instances via checkpointing in the Amazon Elastic Compute Cloud. In Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing (Miami, FL, USA, July 05-10, 2010). CLOUD '11. 236-243.