A Resilient Auction Framework for Deadline-Aware Jobs in Cloud Spot Market

(Poster Abstract)

Abadhan S. Sabyasachi[1], H M Dipu Kabir[2], Ahmed M. Abdelmoniem[1], Subrota K. Mondal[1]
[1]Department of Computer Science and Engineering
[2]Department of Electronic and Computer Engineering
The Hong Kong University of Science and Technology, Hong Kong
{abadhan, amas}@cse.ust.hk, {hmdkabir, skmondal}@connect.ust.hk

***Abstract*—Public cloud providers, such as Amazon EC2, offer idle computing resources known as spot instances at a much cheaper rate compared to On-Demand instances. Spot instance prices are set dynamically according to market demand. Cloud users request spot instances by submitting their bid, and if user's bid price exceeds current spot price then a spot instance is assigned to that user. The problem however is that while spot instances are executing their jobs, they can be revoked whenever the spot price rises above the current bid of the user. In such scenarios and to complete jobs reliably, we propose a set of improvements for the cloud spot market which benefits both the provider and users. Typically, the new framework allows users to bid different prices depending on their perceived urgency and nature of the running job. Hence, it practically allow them to negotiate the current bid price in a way that guarantees the timely completion of their jobs. To complement our intuition, we have conducted an empirical study using real cloud spot price traces to evaluate our framework strategies which aim to achieve a resilient deadline-aware auction framework.**

***Index Terms*—Cloud Spot Market, Cloud Bidding, Online auctions, Deadline-sensitive jobs, Social Welfare maximization.**

1. Introduction

Infrastructure-as-a-Service (IaaS) cloud providers such as Amazon EC2 provides on-demand and scalable that is access to computing resources as service that can be acquired and delivered over the Internet. The IaaS cloud providers' datacenter computing resources allocation must consider both the available datacenter capacity as well as individual user jobs' requirements. With large number of users submitting heterogeneous job requests to cloud datacenter, the resource allocation problem becomes further complicated by creating a highly dynamic environment for the cloud provider.

Recently, Spot pricing by Amazon EC2 is an example of user demand based dynamic pricing strategy [1]. Spot instance pricing create an auction market for allocating the available computing resources (that includes both idle and running spot instances but not from the pool of On-demand or Reserved instances). Users may submit their job requests or bids to acquire spot instance at anytime to the spot market with the help of spot price history. Cloud provider sets the price of each spot instance at regular time slot, which depends on the available computing resources or supply and number of received user bids or demand [2], [3].

Our Contribution, addresses the two most frequent problems faced by the cloud spot market users: (1) how the users bid for spot instances, and (2) what action users take for out-of-bid spot instance failures? *First*, cloud spot instance pricing by the cloud provider proportionate to the dynamic user demand with heterogeneous resource requirements varying over time. While meeting this challenge, cloud provider's objectives are to maximize its revenue and datacenter capacity utilization but also the users' aggregate utility or social welfare maximization of over all cloud spot market system [4], [5]. *Second*, to develop dynamic scheduling algorithms for cost effective, fast and reliable execution of fault-tolerant deadline-aware applications such as HPC jobs, batch jobs and scientific applications using spot instances [6], [7].

Therefore, we have applied the above general cloud scheduling models to the more accurate model of spot instance pricing and user bidding strategy. Our spot market model is not only based on provider's revenue maximization but also considers the following criteria, (1) providing *failure resiliency* of user jobs running in the spot instances with minimum execution overhead and (2) incorporating *soft deadlines* for those user's whose goal is to complete a job within a predefined time with some probability.

2. Deadline-Aware Cloud Spot Market Model

In cloud spot market the users submit job requests or bids for acquiring spot instances to the cloud provider. The received bids are placed in the provider's job queue until the requested spot instances are available. Cloud provider first determines the price of spot instance based on the user demand for the time slot t. The cloud provider allocates based on the number of bids received for time t. If the number of bids waiting in the job queue is high then the demand for that spot instance increases. To handle this rising level of job arrivals in the queue the cloud provider adopts

a dynamic pricing policy for allocating spot instances. The provider rises the spot price for the next time slot $t+1$ in such a way that only those jobs whose bid is higher than the spot price are provided with a spot instance. The number of requests arriving and size of the queue at the cloud datacenter is limited by the Lyapunov drift [3].

2.1. Dynamic Spot Instance Pricing Model

Let us consider a discrete time series $\{1, 2, ...\}$. At each time slot $t \in \{1, 2, ...\}$, we denote the spot price of an instance as $\psi(t)$, the price of an on-demand instance of same type as $\overline{\psi}$ and the provider's minimum cost of running the spot instance as $\underline{\psi}$. We assume that the spot price of an instance doesn't exceed the on-demand price of the same instance type by imposing the constraint $0 \leq \underline{\psi} \leq \psi(t) \leq \overline{\psi}$. At time t, let $\mathcal{D}(t)$ denote the total number of received bids (the demand for a spot instance), $\mathcal{A}(t)$ denote the number of accepted bids or the total system workload (that are higher than the current spot price $\psi(t)$), and $\mathcal{R}(t)$ denote the number of rejected bids (that fall below the current spot price $\psi(t)$). Each successful bidder is charged only current spot price $\psi(t)$ irrespective of her/his actual bid $\hat{\psi}(t)$. The unsuccessful user bids may either wait in the provider's job queue to be reconsidered for the next time slot (called *Persistent bids*), or shall simply quit without any waiting (called *One-time bids*) as shown in figure 1.

At time slot t, assuming that the user bids received by the cloud provider $\mathcal{D}(t)$ follow a uniform distribution f_b which is required to formulate the cloud provider's optimization problem, can be expressed by instance prices as $f_b(x) = \frac{1}{\overline{\psi}-\underline{\psi}}$. Then the number of accepted bids $\mathcal{A}(t) = (\overline{\psi} - \psi(t)) f_b(x) \mathcal{D}(t) = \frac{\overline{\psi}-\psi(t)}{\overline{\psi}-\underline{\psi}} \mathcal{D}(t)$. At time slot t+1, based on the spot price at time t, the number of jobs in the queue may vary which help derive the spot price.

Social welfare maximization objective can be achieved by the cloud provider which is tenants' aggregate utility expressed as, valuation of winning bids at success rate β - provider's aggregate service cost or $\alpha\hat{\psi}(t) - \underline{\psi})\mathcal{A}(t)$. Which will help the provider to maintain its market reputation and quality-of-service (QoS) improvements. The provider guarantees the QoS requirements of cloud users for both the deadline-sensitive jobs. The cloud provider sets the spot price in time slot t $\psi(t)$ to achieves above essential objectives as follows.

$$\begin{aligned} \underset{\psi(t)}{\text{maximize}} \quad & \psi(t) D(t) \tfrac{\bar{\psi}-\psi(t)}{\bar{\psi}-\underline{\psi}} \\ & + (\alpha\hat{\psi}(t) - \underline{\psi}) \mathcal{A}(t) \tfrac{\bar{\psi}-\psi(t)}{\bar{\psi}-\underline{\psi}} \\ & + \beta \, log\left(1 + D(t) \tfrac{\bar{\psi}-\psi(t)}{\bar{\psi}-\underline{\psi}}\right) \\ \text{subject to} \quad & \underline{\psi} \leq \psi(t) \leq \bar{\psi}. \end{aligned} \tag{1}$$

2.2. Resilient Spot Instance Bidding Strategy

In the previous sections we have seen that the provider's strategy and probable optimization formulas. An individual user can't change those formulas but he can bargain with the provider by setting bids intelligently. This section will discuss about our bidding strategy which is most effective for the users can save their progress within 2 minutes after receiving the termination notice. Nonetheless, they need to take a backup of their works periodically and any unsaved progress will be lost. For example, one user is taking backup with 20 minutes interval and the termination occurs at the 25th minute, the progress of last five minutes will be lost.

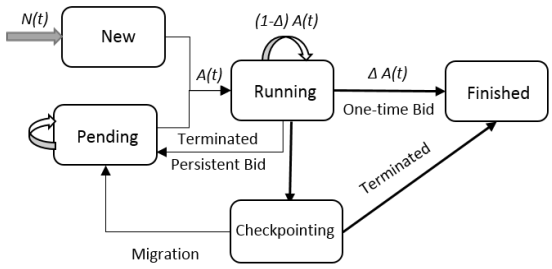

Figure 1. Cloud spot market job queueing model with user bid processing.

Users' spot instance bidding strategy: according to the provider's optimization formulas, a provider set prices that return maximum benefit with little user-consideration. The overall profit is the direct multiplication of profit per user and the number of users. The value of user consideration function always increases with the number of users. When a huge number of the user is bidding at a slightly lower price, both of the profit maximization and user consideration will occur at a slightly lower price. Figure 2(a) is presenting the probability distribution of price after five minutes. Black curve in figure 2(b) is presenting a rough probability distribution of the price after five minutes. The blue curve is presenting the cumulative distribution of probability and the green curve is presenting users intelligent bidding distribution. In order to complete jobs reliably, the user needs to think about the worst case situations too. When the deadline is closer to the minimum remaining job completion time, then the user needs to buy on-demand instances for the reliable completion of the job, probability of which is small.

Mitigating spot instance failures via Checkpointing: as discussed in previous sections, the spot instances are typically perceived as a platform for running non-critical (i.e., elastic non-deadline sensitive) jobs. This perception makes the provider lose a large portion of possible user base and hence revenue. This is mainly because of the bad reputation of spot market instances as an un-reliable service. Here, we discuss the tradeoffs and obstacles that stand against the adoption of spot market as the basis for many time-critical applications on the market [7]. In order to make the spot market an appealing choice for these jobs with soft deadlines, an efficient bidding scheme is not sufficient because it does not give any guarantees. Hence, we advocate a mechanism that involve checkpointing whenever hard-deadlines are approached. To enable such scheme in practice, there are several modifications that could be adopted by existing IaaS cloud spot market to improve its reliability of users' jobs particularly the failure-tolerant Batch processing

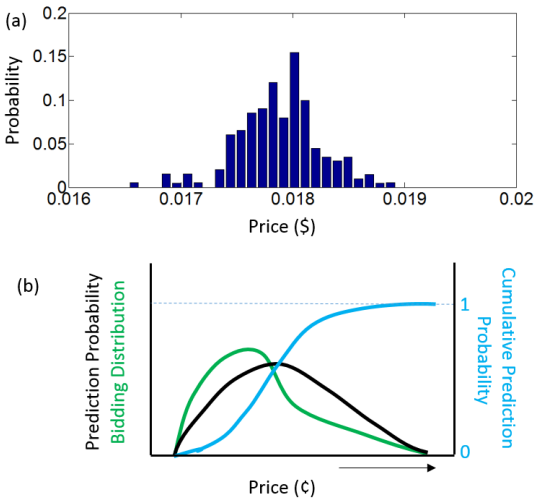

Figure 2. Graphical Representation of cloud bidding strategy. (a) Probability distribution of spot prices. (b) Cumulative probability distribution and bidding strategy.

jobs, MapReduce jobs and scientific workflows [6].

To address the above changes, we developed a Failure and Deadline Aware (FDA) checkpointing algorithm. The algorithm gets as input: the job to decide its checkpoint feasibility, the predication outcome of the bidding strategy, the failure probability distribution related to this job, and the expected recovery time after spot instance interruption. First the checkpoint strategy is chosen based on a decision function that takes into account all four inputs and a corrective function inspired by unsupervised learning techniques.

3. Simulation Results

Probability distribution of Amazon EC2 spot prices: we obtained the cumulative probability distribution of the next sample using the correlation based prediction method. Previously we used that correlation based technique for the determination of the prediction intervals [8]. Individual users can choose that method or other any method for determining the cumulative probability distribution. Prior to applying the theorem, we need to convert the spot price chart to a uniformly spaced samples of 5-minutes interval.

Predictability of Amazon EC2 spot prices: the predictability of spot instance price depends on the success rate of the prediction interval. When the spot price is predictable, discarding 5% less relevant regions from the corners will provide more than 95% success. In the most unpredictable scenario, the PI coverage will be less than 80% while discarding 5% less relevant regions. The correlation based prediction interval formula is previously used in the prediction of power generation and demand. With the current spot price samples, we received a PI coverage of 94.07%, which shows that the spot price is quite predictable on that region. However, the predictability changes between regions. Figure 3 shows the spot prices with point prediction and prediction interval which help users to make better bidding decision by understanding the skewness of probability distribution.

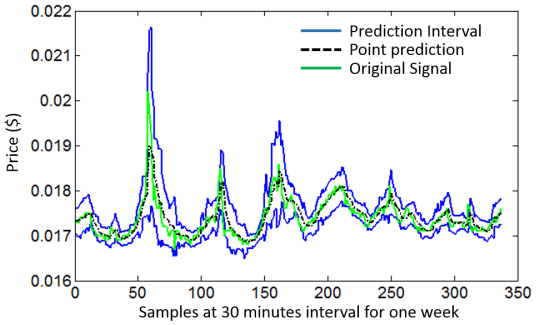

Figure 3. Amazon EC2 C3.large spot instance price of one week during 10-04-2016 to 17-04-2016 with the prediction interval.

4. Conclusion

We have shown the pricing of Amazon cloud spot instances based on the provider's objective in setting of the prices, which clearly indicate there is a correlation between the revenue or profit maximization and social welfare or user's utility maximization. Also we have discussed about the users' to complete jobs within a soft deadline constraint and in a resilient price through such unreliable spot market system. We have proposed improvements from both of the provider's and user's point of views. While the users are suggested to bid at different prices based on their nature and urgency of the job so that they can both negotiate and finish their job in time. By employing real-world cloud traces derived from Amazon EC2 spot price history we have evaluated our resilient spot market pricing and user bidding strategies with appropriate simulation results.

References

[1] "Amazon ec2 spot instances," Jun. 2017. [Online]. Available: https://aws.amazon.com/ec2/spot/

[2] O. Agmon Ben-Yehuda, M. Ben-Yehuda, A. Schuster, and D. Tsafrir, "Deconstructing amazon ec2 spot instance pricing," *ACM Trans. Econ. Comput.*, vol. 1, no. 3, pp. 16:1–16:20, Sep. 2013.

[3] P. Wang, Y. Qi, D. Hui, L. Rao, and X. Liu, "Present or Future: Optimal Pricing for Spot Instances," in *IEEE 33rd International Conference on Distributed Computing Systems*, 2013.

[4] L. Zheng, C. Joe-Wong, C. W. Tan, M. Chiang, and X. Wang, "How to Bid the Cloud," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 5, pp. 71–84, Aug. 2015.

[5] H. Zhang, H. Jiang, B. Li, F. Liu, A. V. Vasilakos, and J. Liu, "A Framework for Truthful Online Auctions in Cloud Computing with Heterogeneous User Demands," *IEEE Trans. Comput.*, vol. 65, no. 3, pp. 805–818, mar 2016.

[6] S. Yi, A. Andrzejak, and D. Kondo, "Monetary Cost-Aware Checkpointing and Migration on Amazon Cloud Spot Instances," *IEEE Transactions on Services Computing*, vol. 5, no. 4, pp. 512–524, 2012.

[7] S. K. Mondal, A. S. Sabyasachi, and J. K. Muppala, "On Boosting Cloud Service Dependability through Optimized Checkpointing," in *15th Int. Symp. Parallel Distrib. Comput.*, 2016.

[8] H. M. D. Kabir, A. Hosen, S. Nahavandi, and A. Khosravi, "Prediction Interval with Examples of Similar Pattern and Prediction Strength," in *The 30th Annual IEEE Canadian Conference On Electrical and Computer Engineering, CCECE 2017.*