Cloud Capacity Planning and HSI based Optimal Resource Provisioning

Naidila Sadashiv*
Department of Computer Science and Engg.
Acharya Institute of Technology,
Bangalore,
India 560 107
*Corresponding Author: Email- sadashiv@acharya.ac.in

Dilip Kumar S M
Department of Computer Science and Engg.,
University Visvesvarya College of Engg.,
Bangalore,
India 560 001

R. S. Goudar
Redknee,
Bangalore,
India 560 045

***Abstract*- Cloud service providers offer spot instances through highest bidding plans that are at a very economical price compared to other pricing plans, namely on-demand and reservation. The usage of spot instance enables utilization of idle resources and provide service for cost sensitive tasks. However, this approach introduces the problem of cloud capacity allocation to different pricing plans that will have impact on the task completion time. To address these issues and improve the providers revenue, in this paper a capacity planning has been carried out based on the prediction of resource requirements for each of the different resource pricing pools. The paper also presents a solution to overcome the burden faced by the service provider due to the free issue of last hour at the time of out-of-bid situation. Simulation carried out based on capacity planning along with hybrid spot instance using Amazon EC2's price show that the resource utilization is improved across the different resource pricing pools with increased number of task completion and improved provider's revenue.**

Keywords- Cloud Computing; Capacity Planning; Resource Provisioning; Prediction; Hybrid Spot Instances;

I. INTRODUCTION

Cloud computing a promising model based on technology and business, has revolutionized the resource usage model. In this paradigm, resource and service requests can be made dynamically or can be booked in advance to have a guaranty of the resource availability. The use of such pricing plans introduces revenue maximization tradeoffs to cloud service providers. Cloud capacity for each of the pricing plans has to be carefully allocated to serve the requests. The issue arises due to the uncertainty in resource requirements under on-demand pricing plan though this model generates highest revenue. In case for reservation category, the requests is certain but the revenue generated is less when compared to resource instance usage under on-demand plan. Unutilized resources is an over head that lead to minimized revenue. For better utilization of the idle resources and boost the profit, the pioneer cloud service provider Amazon introduced spot instances (SI) [1] and preemptible virtual machines [6] at fixed pricing was recently launched by Google Compute Engine. SI pricing model is based on the bidding strategy. User's bid for the spare instances and is allocated provided the bid price is more than the current price of SI. During the resource usage, if the spot price is higher than the user's bid price, the resource usage is abruptly terminated with no reliability of task completion. Such a scenario is referred as out-of-bid. This trade-off that exists between SI cost and service reliability is due to the uncertain request in the cloud environment. To address this trade-off, there exists some work in the literature that discuss about fault tolerant techniques [8], [17], [18], [21]. They include task migration, replication, resubmission and checkpointing. These approaches however impose cost, time and resource overhead on the provider [4], [20] and thereby over rule the reason of using SI. With the growing demand for SI from different type of applications, provides should address the prevailing trade-off between cost and reliability of service [9], [10], [17], [22]. Thus, strategies that optimally utilize the resource by providing reliable service and also improve the provider's profit is essential.

In this paper, the main objective is to carry out capacity planning considering the dynamic nature of user's request to achieve optimal resource utilization and provide uninterrupted service for long running user's tasks. The work also considers hybrid spot instance (HSI) to overcome the overhead involved during the free issue of SI. In summary, the contributions are formulating the capacity planning problem that leads to maximization of SI provider's revenue. Queuing theory based prediction of resources forms the basis to control the admission of requests for the different pools. Proposed strategies are evaluated through simulation driven by the real price traces. Results demonstrates that the capacity planning based HSI improves utilization, revenue and throughput. The rest of this paper is organized as follows. Review of the related existing work and their limitations are highlighted in Section II. Section III discusses about SI in the Amazon Elastic Compute Cloud (EC2). Section IV presents the proposed capacity planning for different resource pools. Section V presents the simulation results with discussion. Lastly, Section VI presents the concluding remarks and scope for future work.

II. RELATED WORK

The section here discusses some of the research works related to capacity planning and about allocation of SI in cloud environment. Adelal et al. [14] presented a framework for revenue management through capacity control for allocating resources to customers. Considering the requests for on-demand and usage level under reservation capacity, stochastic dynamic programming technique based admission control was performed. Anandasivam et al. [2] based on bid-price control technique performed capacity management. Incoming requests were accepted or denied on the basis of bid value to increase revenue. However, the other pricing plans were not considered in these works. Segmentation of capacity among on-demand and SI market based on Markov decision was proposed by Wang et al. [19]. For spot market, an optimal mechanism considering on-demand and SI requests using an auction based pricing was

978-1-5090-3239-6/17/$31.00©2017IEEE

considered, assuming that reserved requests will be satisfied. Their work differs from the proposed work since reliability of task completion is given preference in the current work by minimizing abrupt task termination. To handle the dynamicity in the workload during limited resource capacity, Rodrigo et al. [3] proposed a prediction based proactive approach for dynamic provisioning of resources for SaaS applications. The ARIMA based prediction model was considered to improve the utilization and response time for users. A framework for allocation in cloud was presented by Verma et al. in [16]. Here, based on the dynamic nature of resource requirements, service tenants were classified and its resource prediction was prioritized to minimize the prediction time. Toosi et al. [15] proposed an auction with greater probability of truthfulness to improve provider's profit. Their approach does not depend on the history of bidding process. These all works considered the prediction model as in the current approach however, capacity planning and different pricing plans were not focused. Chenhao Qu et al. [12] have explored the utilization of spot instances to provision the availability-critical web applications by taking the advantage of differences in spot instance prices to reach improved availability and cost saving. Deepak et al. [11] presented an on-demand and adaptive spot based
just in time scheduling algorithm for scientific workflow to provide fault tolerant schedules. The algorithm considered the different pricing of resources to minimize the time and cost. In an similar direction, an approach to estimate the spot instance prices was proposed in [17]. Techniques for handling the fault such as task migration, task duplication and its analysis with checkpointing were carried out. This approach however imposes cost over head and is addressed through HSI strategy in the current work. Sangho et al. [21] proposed an approach to minimize monetary costs by using SI. Different static and dynamic checkpointing strategies were studied and analyzed.
These above mentioned works considered SI for serving the cost sensitive applications, focused on fault techniques and to provide reliable service however, not in a unified manner. Lack of capacity planning and addressing the free issue of SI resource in the literature motivates the current work as its plays a vital role in revenue maximization.

III. SI IN AMAZON ELASTIC COMPUTE CLOUD

This section highlights spot instances and its characteristics. As on date, Amazon renders different type of instances across 11 different regions. Each instance is a combination of different resources that include CPU, memory, I/O, disk etc. [1]. Instances. Among the instances provided, few instances are general purpose, and some are grouped as compute, memory and storage optimized for running requirement specific applications. The instances are configured with special features that enables application deployment, management, and scalability of applications. These instances are allotted to users on the basis of on-demand or through reservation. The left over instances that are unallocated and idle are provisioned as SI. It follows dynamic bid pricing model as it depends on the uncertain user's request. SI price is freely available and sample of it is shown in Fig. 1. Amazon has also launched a fleet of spot instances, which represents a collection of SI that work together as part of a distributed application. Spot fleet is responsible for resource discovery, resource bid management and also running their workloads at nominal possible price. Upto 1000 spot fleets with 3000 instances per fleet and per region is allowed. Details regarding the target capacity, maximum bid price and dates are essential to make a request. Few important characteristics of SI are given as below:

- SI is provisioned when user bids a price that is greater than current SI price and are hourly charged.
- SI is abruptly terminated provided the new spot bid price
  is more than the current users bid price.
- At the time of out-of-bid situation, last hour is given for free however, will be charged for the whole hour if termination is requested from the user.
- Specification of spot instances for a predefined duration
  workload is allowed however, there exists no reliability
  for QoS.

In order to provide the benefits of SI and to address the tradeoff
between cost and reliability of service, several checkpointing and fault tolerant strategies are been performed.

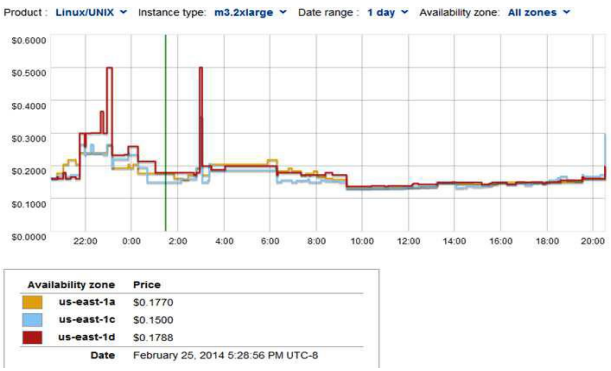


Fig. 1: Amazon SI Bidding

IV. PROPOSED CAPACITY PLANNING

To utilize the resource and maximize revenue, it is vital for the CSP to target for capacity planning among the different pricing plans. This section describes an approach for capacity allocation decision such that overall revenue is increased with minimized task abrupt termination.
Assume the total provider's capacity as C for a given type of instance. Currently, some resources are allocated to user's requests under the three pricing policies that include on-demand, reservation and SI and is denoted as R_OD, R_R *and* R_SI respectively. Based on the historical requests made at the capacity pool, prediction for on-demand resources P_OD and reservation resources P_R are been carried out in Section IV-1. On the basis of this prediction based computation and allocation, the remaining capacity can be allotted for running the spot instance requests as given below:

$$P\_SI = C - (R\_OD + R\_R + R\_SI + P\_OD + P\_R) \quad (1)$$

Utilization of resources through this capacity planning aims to maximize revenue and is defined as below:

$$Maximize \sum (R\_OD_t + P\_OD_t) * P + (R\_R_t + P\_R_t) * \alpha * P + (R\_SI_t + P\_SI_t) * \beta$$

where *t=0 to T-1* (2)

subject to constraint

$R\_OD_t + R\_R_t + R\_SI_t + P\ OD_t + P\ R_t + P\ SI_t \leq C$

where t=0....., T-1

α and P denotes on-demand resource price and β represents the SI price. The above constraint guaranties that the ongoing on-demand instances, reservations and SI along with the predicted capacity need remains within the providers capacity. This ensures that no SLA violations occurs for the on-demand and reservation contracts.

1) *How Much to Provision: Modeling the Capacity Planning:*

The goal of the capacity planning is to allocate the required capacity to the different pricing plans so that resource is efficiently utilized with less abrupt SI termination. To address this issue, the capacity pool is modeled as G|G|1 model that generally captures arbitrary distribution of arrival and service times. Based on the maximum incoming input request rate obtained from the historic information and the capacity of the server, prediction of on-demand resources are been computed. The model assumes that all the instances are homogeneous however, can be enhanced to incorporate more than one type of instances. On-demand resource instances predicted for on-demand capacity pool is given in Equation (3).

$$P\_OD_t = (\lambda_{aod} * \phi) / \lambda_{cod} \quad (3)$$

where λ_{aod} represents an estimate of arrival rate distribution seen by the on-demand capacity pool. The rate of request that can be served by a single on-demand instance is represented as λ_{cod} and is obtained from queuing theory result [7] as given in Equation (4).

$$\lambda_{cod} \geq [\ st_{od} + (\sigma^2_{aod} + \sigma^2_{sod})\ /\ (2 * (\ \Upsilon\ - st_{od}))\ ]^{-1} \quad (4)$$

where Υ represents the expected mean response time of instance, and st_{od} represents the average service time for a request from the instance in on-demand capacity pool. σ_{aod} and σ_{sod} denotes the variance of inter-arrival and inter-service time respectively.

A. Capacity Planning based HSI Provisioning

Capacity planning using HSI resource provisioning is presented in this section. The aim of HSI based resource provisioning is to improve the reliability of current SI users by prohibiting abrupt termination during out-of-bid situation. This is performed by enabling users to stretch their bid price till checkpointing is carried out or to perform rebid process until the user's bid price equals the on-demand price [13]. Such a bidding approach is enforced in ebay as a measure to avoid rebidding.

Algorithm 1: Capacity Planning based HSI

Data: Total Capacity C, active-SI, usr-price, sb-price, sb-flag, sb-till-od, od-price, HSI-flag, sp-price, Another-chance=1

Result: usr-price

```
1   R_Util=Compute-Capacity-Utilized();
2   P_OD= Expected-Ondemand-Resource();
3   P_R= Expected-Reserved-Resource();
4   //Capacity planned for spot instance resource pool;
5   SI=C-(R_Util+P_OD+P_R);
6   if Out-of-Bid AND HSI then
7     while HSI-flag do
8       if old-user k Another-chance then
9         Another-chance = 0 ;
10        if sb-flag then
11          if usr-price ≤ sb-price and sp-price ≤
            sb-price then
12        Update-Hybrid-Spot-Price();
13        Continue;
14      else
15        terminate-si() ;
16    else if sb-till-od then
17      if usr-price _ od-price then
18        Update-Hybrid-Spot-Price();
19      else if usr-price ≥ od-price then
20        sp-price-equals-od-price++ ;
21        usr-price=od-price;
22        break ;
23  else
24    terminate-si() ;
25 else
26 terminate-si() ;
```

HSI strategy enables uninterrupted task completion within the expected time. This strategy forbids abrupt task termination and the burden of checkpointing. HSI will benefit the task that is nearing its execution completion. The work here considers Amazon spot instances however, it can be extended for instances from other service providers.

V. PERFORMANCE EVALUATION

This section discuses about the conduction of three different groups of experiments. First, the overhead of free issue of last hour and its cost is evaluated. Secondly, capacity planning based resource management framework is analyzed. Lastly, performance of the proposed capacity planning based hybrid spot instance is compared with other baseline approaches using trace-driven simulations.

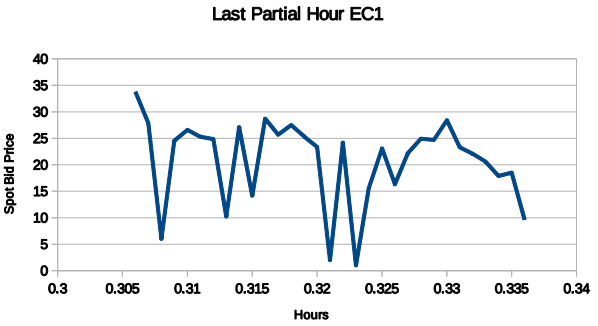


Fig. 2(a): Out-of-bid Overhead in Hours

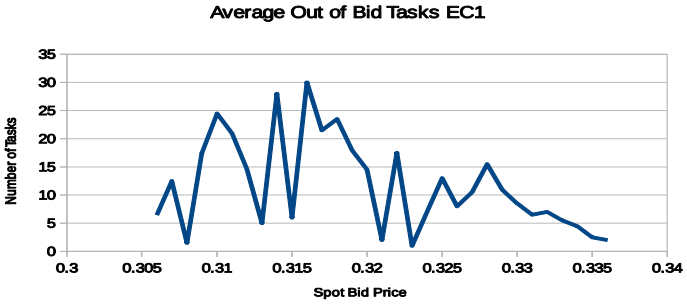


Fig. 2(b): Out-of-bid Overhead Count

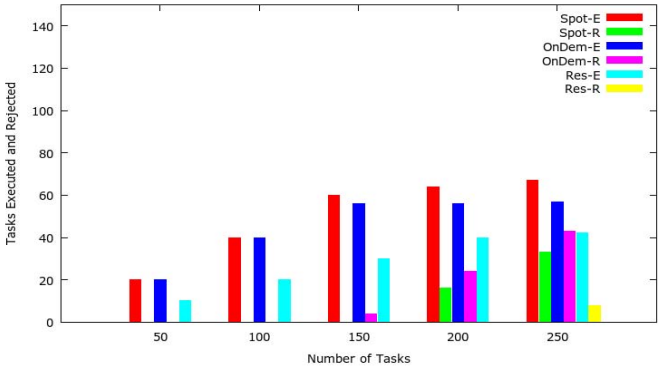
Most of the cloud service providers often regard their workload traces as confidential. Google has published a dataset related to its general workload [5]. Using this as the basis, requests are synthesized by normalizing the requests time requirement to the longest lifetime in the traces. Requests are categorized on the basis of the type of request made. Requests that are non fault tolerant are provisioned with on-demand instances, long term requests requirement are fixed with reservation and for requests to be performed at low compute price for a short duration are assigned spot instances. On this basis, the requests are labeled with one of these pricing plans randomly with the help of Gaussian distribution considering the sample price of Amazon EC2 Instances [1]. For simulation, 100 heterogeneous applications of different sizes and completion time that range between 1 - 1000 minutes are considered. The parameters such as out-of-bid, tasks completed count, resource utilization and price involved for UNIX/Linux m1.small (EC 1) are considered.

A. SI Overhead

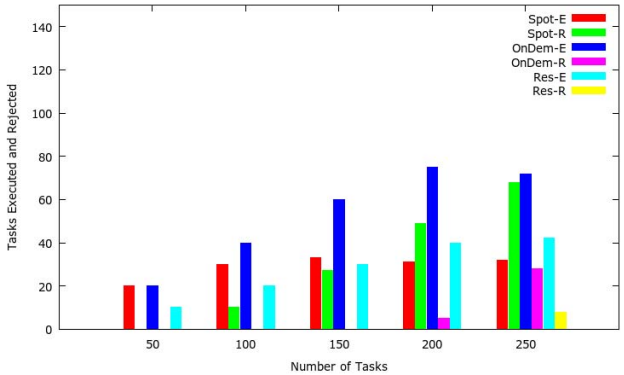
Amazon leverages a pricing model that performs the resource usage charging on an hourly basis. However, during out-of-bid situation the last few minutes of an hour is been let-off and is not charged. Fig. 2(a) and 2(b) demonstrates the over head in terms of hours and the out-of-bid count involved during the simulation of 100 applications.

B. Impact of Capacity Planning

Capacity planning has been carried out to improve the revenue by optimally allocating the capacity among the three different pricing resource pools that include on-demand, reservation and SI.



3 (a) Task Execution and Rejection without Capacity Planning



3 (b) Task Execution and Rejection with Capacity Planning

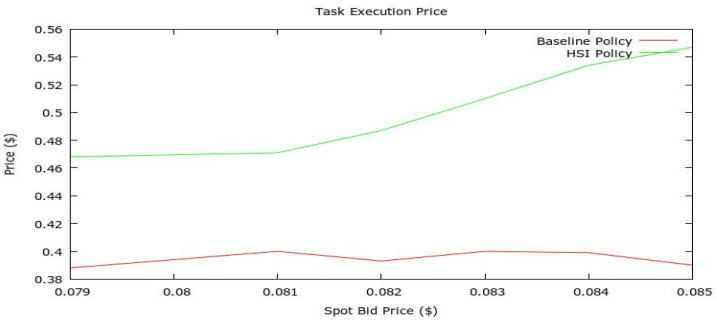


Fig. 4: Comparison of Providers Revenue based on Capacity Planning

1) Analysis of Task Execution and Rejection: On the basis of historical information from the capacity pool, each of the pricing plans are assigned with capacity such that the number of requests executed are maximized and rejection of requests are reduced in all the three pricing plans. Fig. 3(a) presents the number of tasks executed and rejected without capacity planning in all the three types of resource pools and Fig. 3(b) demonstrates the impact of capacity planning based on prediction. In Fig. 3(a), more resource is allocated for on-demand pricing pool without considering the expected requirement. This results in execution of on-demand requests however, few spot requests are fulfilled with large number of spot request being rejected. On the other hand, based on the predicted requests when capacity is planned it is seen that maximum number of on-demand and as well as spot request's are executed. When the number of request tasks is more than 200, some of the on-demand tasks as seen in Fig. 3(b) will be delayed as spot instances will be suspended to overcome the shortage of resource capacity. The number of such tasks is however less in Fig. 3(b) than in Fig. 3(a).

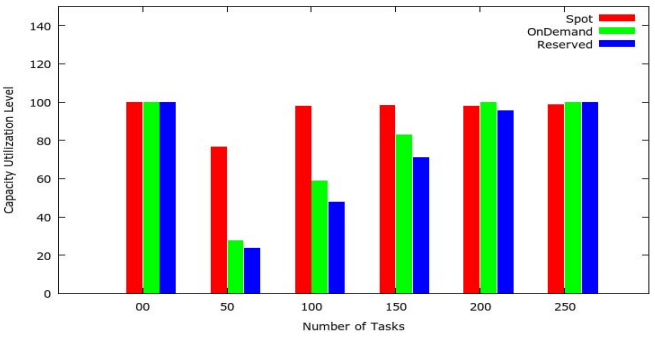


Fig. 5 (a): Resource Utilization without Capacity Planning

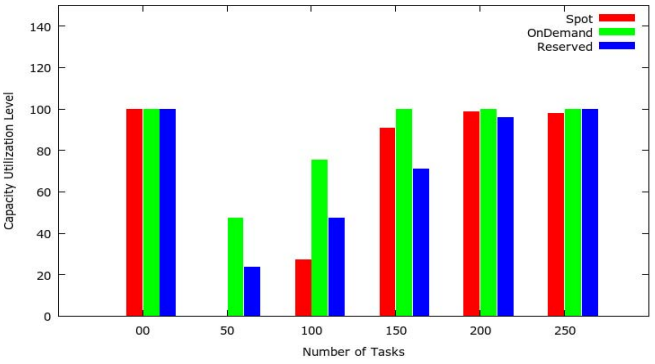


Fig. 5 (b): Resource Utilization with Capacity Planning

2) Analysis of Revenue based on Capacity Planning:

With capacity planning, sufficient amount of resources are allocated for on-demand and reservation pool, the rest of the total capacity is allocated to spot instances. Hence, situation of lack of

resource faced by on-demand request is very less that results in very minimal task interruption leading to higher throughput of spot instance. This minimizes the last partial hour overhead presented in Fig. 2. Thus, revenue obtained by the provider is more when capacity is planned as demonstrated in Fig. 4.

3) Analysis of Resource Utilization:
Utilization of resource when capacity is not planned and while capacity planning is considered are presented in Fig. 5(a) and Fig. 5(b) respectively. It is seen that resources are not fully utilized in Fig. 5(a) and some spot requests are being rejected. Appropriate planning leads to better utilization as seen in Fig. 5(b). The result however, depends on the precision of prediction model. Results revels that there is still large amount of resources left under on-demand resource pool in Fig. 5(b) and this ensures service availability for the new requests without affecting the tasks running under HSI.

C. Impact of HSI and Comparison with Baseline Policies:
The impact of capacity planning based HSI strategy on the task performance and its comparison with baseline policies that include base, hourly and no checkpointing has been performed. For base checkpointing, the checkpointing is performed just-in-time. Whereas for hourly checkpointing, it is done at the end of every hour and is not applicable for nocheckpointing approach. Different size applications are simulated on various type of instances. Results of this simulation are discussed as given below:
Number of Executed Tasks: HSI enables the uninterrupted task execution by stretching the bid or through redefinition of user bid price. This increases task throughput when compared to other existing policies where the tasks are abruptly terminated as presented in Fig. 6 (a). Such tasks are a burden for both the user and the providers in terms of resource and time.
Tasks Execution Time: The execution time for 100 tasks is simulated as 500 minutes each. Based on the working of HSI, abrupt termination is forbidden that lead to timely completion of the HSI based tasks. This is not the case for existing policies and hence, the terminated tasks have to be restarted by again going through the bidding process. This results in longer execution time for baseline policies as seen in the Fig. 6 (b).
Task Execution Cost: Task execution cost on m1.small (EC 1) is presented in Fig. 6 (c). The cost involved for the task completion is very less for HSI approach when compared to baseline policies. As discussed above, the rebidding involved due to out-of-bid situation causes an increase in the cost in case of the baseline policies.

VI. CONCLUSION

Amazon has pioneered spot instance as a resource provisioning model that delivers unallocated idle resources through highest bidding strategy. The aim of SI is to improve the provider's revenue and render service to cost sensitive users. In this paper, a capacity planning approach is presented that identifies the capacity pool size for the pricing plans that include on-demand, reservation and SI that leads to improved providers revenue. Capacity planning based on prediction and HSI resource provisioning to achieve optimal resource utilization is the foundation of this work. Simulation results demonstrate that the proposed strategy improves reliability followed by throughput through stretch and rebid approach during out-of-bid situation. Results also show that the cloud SI provider's revenue can be optimized by preventing the free release of last incomplete service hour and checkpointing through HSI. Future plan is to integrate a reactive module as the next level of capacity planning. Another aspect is to carry out the current work on various other hosted services other than EC2 and analyze its impact on the QoS.

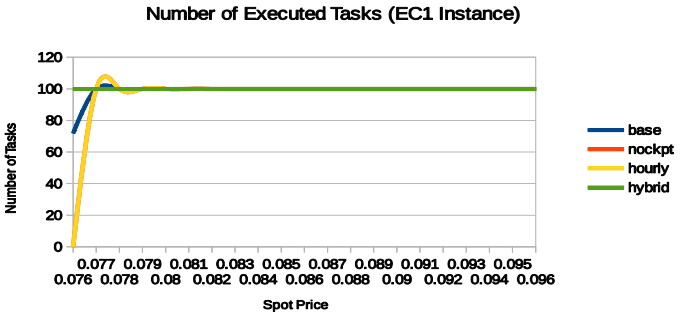


Fig. 6 (a): Number of Tasks Executed

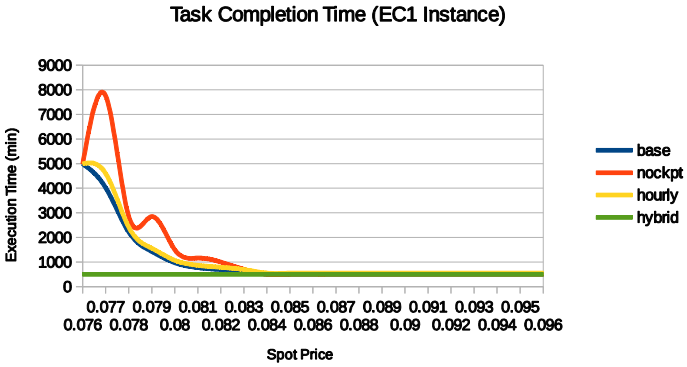


Fig. 6 (b): Tasks Execution Time

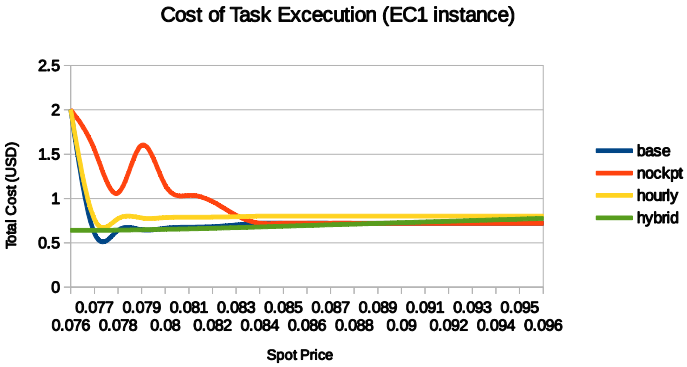


Fig. 6 (c): Task Execution Cost

REFERENCES

[1] Amazon. http://aws.amazon.com/ec2/spot-instances, (accessed May, 2016).
[2] A. Anandasivam, S. Buschek, and R. Buyya. A heuristic approach for capacity control in clouds. In 2009 IEEE Conference on Commerce and Enterprise Computing, pages 90–97, 2009.
[3] R. N. Calheiros, E. Masoumi, R. Ranjan, and R. Buyya. Workload prediction using arima model and its impact on cloud application's qos. IEEE Transactions on Cloud Computing, 3(4):449–458, 2015.
[4] K. Chard and K. Bubendorfer. High performance resource allocation strategies for computational economies. IEEE Tran. on Parallel and Distributed Systems, 24(1):72–84, Jan 2013.
[5] Google. http://code.google.com/p/googleclusterdata, (accessed February, 2014).
[6] Google. http://code.google.com/p/googleclusterdata, (accessed July, 2016).
[7] Leonard Kleinrock. Queueing Systems, volume II: Computer Applications. Wiley Interscience, 1976.
[8] Haikun Liu, Hai Jin, Xiaofei Liao, Chen Yu, and Cheng-Zhong Xu. Live virtual machine migration via asynchronous replication and state synchronization. IEEE Tran. on Parallel and Distributed Systems, 22(12):1986–1999, Dec 2011.
[9] Samaan N. A novel economic sharing model in a federation of selfish cloud

providers. IEEE Tran. on Parallel and Distributed Systems, 25(1):12–21, Jan 2014.
[10] M. Spreitzer M. Steinder A. Tantawi N. Chohan, C. Castillo and C. Krintz. See spot run using spot instances for mapreduce workflows. In 2nd USENIX Conf. on Hot topics in Cloud Computing, pages 1–7, June 2010.
[11] Deepak Poola, Kotagiri Ramamohanarao, and Rajkumar Buyya. Enhancing reliability of workflow execution using task replication and spot instances. ACM Transactions on Autonomous and Adaptive Systems (TAAS), 10(4):30, 2016.
[12] Chenhao Qu, Rodrigo N. Calheiros, and Rajkumar Buyya. A reliable and cost-efficient auto-scaling system for web applications using heterogeneous spot instances. Journal of Network and Computer Applications, 65:167 – 180, 2016.
[13] N. Sadashiv, D. Kumar S M, and R. S. Goudar. Hybrid spot instance based resource provisioning strategy in dynamic cloud environment. In 2014 International Conference on High Performance Computing and Applications (ICHPCA), pages 1–6, 2014.
[14] A. N. Toosi, K. Vanmechelen, K. Ramamohanarao, and R. Buyya. Revenue maximization with optimal capacity control in infrastructure as a service cloud markets. IEEE Transactions on Cloud Computing, 3(3):261–274, 2015.
[15] Adel Nadjaran Toosi, Kurt Vanmechelen, Farzad Khodadadi, and Rajkumar Buyya. An auction mechanism for cloud spot markets. ACM Transactions on Autonomous and Adaptive Systems Journal, 11(1), February 2016.
[16] Manish Verma, G. R. Gangadharan, Nanjangud C. Narendra, Ravi Vadlamani, Vidyadhar Inamdar, Lakshmi Ramachandran, Rodrigo N. Calheiros, and Rajkumar Buyya. Dynamic resource demand prediction and allocation in multi-tenant service clouds. Concurrency and Computation: Practice and Experience, pages 1 – 14, 2016.
[17] W. Voorsluys and R. Buyya. In IEEE 26th International Conf. on Advanced Information Networking and Applications.
[18] S. Garg W. Voorsluys and R. Buyya. Provisioning spot market cloud resources to create cost-effective virtual clusters. In 11th international conference on Algorithms and architectures for parallel processing, pages 395–408, Feb 2011.
[19] W. Wang, B. Li, and B. Liang. Towards optimal capacity segmentation with hybrid cloud pricing. In Distributed Computing Systems (ICDCS), 2012 IEEE 32nd International Conference on, pages 425–434, 2012.
[20] Yi-Min Wang, Yennun Huang, Kiem-Phong Vo, Pe-Yu Chung, and C. Kintala. Checkpointing and its applications. In 25th International Symposium on Fault-Tolerant Computing, pages 22–31, June 1995.
[21] Sangho Yi, D. Kondo, and A. Andrzejak. In IEEE 3rd International Conf. on Cloud Computing.
[22] Jianfeng Zhan, Lei Wang, Xiaona Li, Weisong Shi, Chuliang Weng Wenyao Zhang, and Xiutao Zang. Cost-aware cooperative resource provisioning for heterogeneous workloads in data centers. IEEE Tran. on Computers, 62(11):2155–2168, Nov 2013.