



Cloud Resource Allocation Based on Historical Records: An Analysis of Different Resource Estimation Functions

Qi Hu, Mohammad Aazam, and Marc St-Hilaire^(✉)

Department of Systems and Computer Engineering,
Carleton University, Ottawa, ON, Canada
qihu3@cmail.carleton.ca, aazam@ieee.org, marc.sthilaire@carleton.ca

Abstract. Resource allocation is an important problem for all Cloud Service Providers (CSPs). Some recent studies propose interesting resource assignment models based on the historical behavior of customers. However, they have a few limitations. For example, some of the proposed models are not suitable in all situations or server load conditions. In this paper, we address such limitations from the model in [1] and introduce several new resource estimation functions to achieve better resource allocation. More precisely, four new mathematical models are first proposed and analyzed. Then, we used the CloudSim simulation toolkit to compare the mathematical results and the simulation results. Our preliminary analysis indicates that different models should be used for different situations in order to achieve better resource utilization.

Keywords: Cloud computing · Resource management
Resource allocation · Resource estimation · Mathematical analysis
Simulation · CloudSim

1 Introduction

Nowadays, the number of large-scale data centers is growing significantly and the complexity of the network infrastructure is also increasing. More importantly, the power consumption of those data centers is enormous. In 2010, Google data centers used about 2.26 million megawatt hours of electricity and generated 1.46 million metric tons of carbon dioxide [2]. Additionally, building a data center leads to excessive capital expenditure as data centers are usually built to serve infrequent peak loads resulting in low average utilization of the resources. In cloud computing, resource utilization and power consumption are highly coupled. In fact, data centers are consuming a huge amount of energy even when resource utilization is low. A recent research shows that the resource utilization is lower than 50% on average [3]. Therefore, having dynamic resource management is important as it directly relates to power management.

Cloud utilization can also be impacted by the behavior of customers. In fact, cloud customers have a fluctuating behavior resulting in different loyalty patterns. For example, a somewhat disloyal customer could relinquish resources before the scheduled end time and therefore creating a loss for the cloud provider. According to [4], a Cloud Service Customer (CSC) usually overestimates the amount of the required resources which leads to low resource utilization. Since less time will be taken to complete the task, the CSC may choose to relinquish the service before the service expires. As can be seen, solutions need to be developed to improve resource management in order to reduce unnecessary costs and improve the resource utilization while satisfying the service level agreement. Authors in [1] discuss resource estimation on the basis of customer's historical record. In their model, resources are allocated on the basis of the loyalty of the requesting customer. That is, a loyal customer will get incentive or more resources, while a disloyal customer will be assigned with resources in a cautious way. The end goal is to minimize resource underutilization and save power. However, the main drawback in the proposed work is that it does not take into account server load. We counter this limitation in our work.

It is important to find proper resource estimation mechanisms to increase the server utilization and reduce unnecessary power consumption. In this paper, we propose and analyze different resource estimation functions with the goal of improving resource utilization. More precisely, we want to:

- Propose and analyze (mathematically) four different strategies to assign resources under various criteria such as server utilization, the amount of allocated resources, and the customer experience.
- Implement and simulate the different models in a cloud computing environment such as CloudSim.
- Compare the mathematical results with the simulation results of the different strategies.

The rest of this paper is organized as follow. Section 2 provides an overview of the related work with more emphasis on [1] as the work presented in this paper is an extension from [1]. Section 3 briefly introduces the new models with a mathematical analysis of their pros and cons. Section 4 describes the simulation environment and provides an analysis of the simulation results. Finally, Sect. 5 concludes the paper and outlines future work on the topic.

2 Related Work

Performing analysis on large-scale trace logs is fundamental to deriving realistic models. In [4], Moreno et al. propose an approach to derive realistic workload models. They have found that workloads are various in different observation periods. Moreover, the cloud environment does not show obvious periodical behavior and users tend to overestimate their requirements. Beloglazov and Buyya present a decentralized architecture of the energy aware resource management system for cloud data centers [5]. However, the authors do not mention any details about

the parameters of each policy used in the simulation part such as the value of the threshold and the reason for using that value. In addition, for the virtual machine reallocation problem, there is no explanation on how to determine new placement for virtual machines.

Wang and Yang develop an energy conserving resource allocation scheme with prediction for cloud computing systems [6]. This scheme can predict the trend of coming tasks and their features; and the system can react by shutting down a physical machine (PM) or starting up a new PM according to the trend. Dabbagh et al. propose an integrated resource management framework that reduces energy consumption in cloud data centers [7] based on Google traces. However, they just use one trunk of the traces as the training data set and one trunk as validation data set. Majumdar et al. describe broker-based system on a new framework that performs proactive auto-scaling [8]. The proposed broker-based system determines the number of allocated resources by predicting the amount of requested resources in the future. However, the characteristics of the requests are hard to predict since there are different types of customers with different behaviors. In addition, the relinquishment of services and reimbursement are not considered in the system.

In [1], Aazam and Huh propose a broker-based resource estimation model which depends on the behavior of customers. Instead of assigning resources equally to all users, the model uses the concept of relinquish probability to assign resources. The users' relinquish probability is calculated according to their histories. Thus, the broker will calculate the amount of resources to be allocated to users based on the prediction model and decide the price of the allocated service. Besides, the pricing mechanism of the model is based on giving incentive to more reliable customers to increase loyalty. Based on the resource estimation model, they also add a refunding system with considerations for quality degradation of services in [9]. For completeness and since our work is based on this paper, the main equation for the resource assignment model is summarized below. Please refer to [1] for the complete model and more details.

$$R = \Psi_{pi} \left((1 - \bar{x}(P_i(L|H)_s)) - \sigma^2 \right) (1 - \Omega_i) \quad (1)$$

In the above equation, R represents the required resources which can be CPU, storage or memory; Ψ_{pi} is the basic price of the resource which is typically decided when the contract is negotiated. $\bar{x}(P_i(L|H)_s)$ is the average value of the Service Oriented relinquish Probabilities (SOP) of a particular customer giving up the requested resource. A customer relinquishing resources means that the customer stops using the service before the scheduled end time. For example, a customer may be assigned resources for his service for 10 hr but stops using it after 5 hr. In this case, the relinquish probability for this request is 50%. Since customers can have fluctuating behavior in using resources, the authors use σ^2 as the variance of SOP to determine the actual SOP of each customer. Ω_i , the Average Overall relinquish Probabilities (AOP), is calculated based on the overall history (regardless of the service) of the customer with the cloud

provider. It is different from SOP which only calculates the average relinquish probability of the customer using the same service.

The model introduced in [1] is novel and very interesting. However, we believe the model has some flaws as pointed out below:

1. The model uses the price of the service to differentiate between services. In fact, the broker will allocate a general amount of resources to the customer without specifying what kind of resources (e.g., memory, storage, etc.) is included. We believe that this is not very realistic as real cloud providers have to map the services to real physical resources. According to the model in [1], if two services have the same prices, they are treated as if they were using the same amount of resources even if they are actually two different kinds of services.
2. The authors use the variance of SOP in Eq. (1) to help predict the actual behavior of each customer. However, the variance is a value that indicates how far a set of numbers is spread out so it does not make the results more accurate.
3. The resource estimation model uses the notion of SOP and AOP to decide how much resources should be allocated, where SOP and AOP have the same weight. In fact, we believe SOP should have more weight because it represents the average of the service oriented relinquish probabilities of a particular customer giving up the same resource that is currently being requested.
4. The model will assign resources based on the relinquish probability of the customer. This means that if two customers with the same relinquish probabilities ask for the same service at different point in time, they will always get the same amount of resources. Although this model is better than a flat assignment policy (everyone gets the same amount of resources), we believe it could be made even more dynamic by using different models.
5. Some important parameters are not considered when assigning resources to users. For example, the current cloud utilization and the type of service could have an important impact on the amount of resources that is allocated.
6. Finally, the paper only partially presents the simulation parameters which make it difficult to regenerate similar results. Also, only partial results are presented which does not show the complete picture of how the model behaves.

3 Resource Estimation Modeling

In this section, we present the new formulation to address the weaknesses identified in the previous section. First, we model customer requests to address flaw (1) mentioned above. Then, a new resource estimation function is proposed to tackle flaws (2) and (3). Finally, four different resource estimation functions are proposed and evaluated to address flaws (4) and (5).

3.1 Modeling Customers' Requests

In the new model, the customer's requests are modified as shown in Eq. (2):

$$N = \begin{cases} N_{CPU} \\ N_{Memory} \\ N_{Storage} \end{cases} \quad (2)$$

N means the overall request from a customer including the amount of requested CPU (N_{CPU}), the amount of requested memory (N_{Memory}) and the amount of requested storage ($N_{Storage}$). When a customer submits a request to the broker, the broker will analyze the request and obtain the required amount of resources for delivering the service. A request from a customer is split into three parts at the broker's side: CPU, memory and storage. These types of resources stand for three fundamental characteristics customers pay for with cloud services: compute, data transfer out and storage [10]. They are common resources deployed in several cloud providers such as Amazon Web Services, Dropbox, Microsoft Azure, etc. Since the request is more specific, the whole model will be more realistic.

3.2 Modification to the Resource Estimation Model

The resource estimation model is modified as shown below.

$$R = \sum_{i=1}^n \begin{cases} N * (1 - SOP) * (1 - AOP) \\ 0 \end{cases} \quad (3)$$

$$N, R \in \{CPU, Memory, Storage\}$$

In Eq.(3), N is the requested resources and R represents the allocated resources. SOP is the average of the relinquish probabilities of the currently requested service, AOP is the average of the relinquish probabilities of all services from the same cloud provider. The way to calculate SOP and AOP is the same as explained in [1]. However, this model has some weaknesses. For example, suppose customer 1 ($SOP = 0.1$, $AOP = 0.9$) and customer 2 ($SOP = 0.9$, $AOP = 0.1$) ask for the same service. According to Eq.(3), they will be allocated the same amount of resources. This unreasonable situation happens because SOP and AOP have the same weight. As mentioned previously, SOP should have more weight than AOP because SOP is more related to the current request. Considering this point, the new way to calculate the overall relinquish probability is described by the equation below. As can be seen, the weight of SOP is two times the weight of AOP which makes sure that SOP will have a more important impact when deciding the amount of resources to allocate.

$$x = \frac{2}{3} * SOP + \frac{1}{3} * AOP \quad 0 \leq x \leq 1 \quad (4)$$

In Eq.(4), x stands for the general relinquish probability composed of SOP and AOP . The range of the relinquish probability is between 0 and 1 (instead of 0.1 to 1 as mentioned in [1]) to include all possibilities so that the model will be more comprehensive. When $x = 0$, it means that the customer has never

relinquished resources in the past (i.e. a perfectly loyal user). Coming back to the example above with the new equation, customer 2 will get more resources due to a higher value of *SOP*.

Based on the new equations above, four models will be introduced and illustrated. Then, the models will be compared with respect to different criteria such as the overall amount of allocated resources, the overall resource utilization and the customer’s experience.

3.3 Introduction and Analysis of the Models

When the broker receives a request from a customer and the customer’s record from the provider, it can calculate the amount of resources to be allocated. The general formulation is shown in Eq. (5). The most important part of the formulation is the estimation function (i.e. $f(x)$). The larger the value of $f(x)$ is, the more resources will be allocated. Therefore, by using different estimation functions, different goals could be achieved. Figure 1 and Eqs. (6) to (9) represent the estimation function for models 1 to 4 respectively.

$$R = N * f(x) \tag{5}$$

$$f(x) = (1 - x)^2 \tag{6}$$

$$f(x) = 1 - x^2 \tag{7}$$

$$f(x) = \begin{cases} 0.5 + 0.5 * (1 - 2x)^2 & 0 \leq x \leq 0.5 \\ 0.5 - 2 * (x - 0.5)^2 & 0.5 \leq x \leq 1 \end{cases} \tag{8}$$

$$f(x) = \begin{cases} 1 - 2x^2 & 0 \leq x \leq 0.5 \\ 0.5 * (2 - 2x)^2 & 0.5 \leq x \leq 1 \end{cases} \tag{9}$$

To analyze the four models, different metrics such as the overall amount of allocated resources (*AR*), the overall amount of relinquish resources (*RR*), the amount of utilized resources (*UR*) and the service utilization (*U*) are calculated. Equations (10) to (13) are used to calculate the different metrics.

$$AR = 1 \cdot \int_0^1 f(x)dx \tag{10}$$

$$RR = 1 \cdot \int_0^1 (f(x) \times x)dx \tag{11}$$

$$UR = AR - RR \tag{12}$$

$$U = \frac{UR}{AR} \tag{13}$$

For the calculation, we assume that:

1. The requested amount of resources (N) and the duration of the request are set to 1.
2. The relinquish probability of customers is evenly distributed between 0 and 1.
3. Each customer will relinquish with the same probability as his overall average relinquish probability.

The result of these calculations is shown in Table 1.

Table 1. Mathematical analysis of different models

	AR	RR	UR	U
Model 1	1/3	1/12	1/4	3/4
Model 2	2/3	1/4	5/12	5/8
Model 3	1/2	3/16	5/16	5/8
Model 4	1/2	7/48	17/48	17/24

By analyzing model 1, we can see that it does not allocate a lot of resources to user's requests (AR : $\sim 33\%$). In fact, model 1 is the model that allocates the least amount of resources. Moreover, this model allocates much more resources to loyal users (when $x \leq 0.5$) than disloyal users (when $x > 0.5$), which improves the fairness. Since model 1 allocates the least amount of resources and its utilization is the highest, it could be a good choice when the server is overutilized and is short of resources.

As far as model 2 is concerned, we can see that it allocates much more resources than model 1. In fact, among all the models, model 2 is the one that allocates the most resources ($\sim 66\%$). For customers who become more loyal, the amount of allocated resources increases significantly at first and then slows down. This is a good incentive for disloyal customers to become more loyal. For loyal users, the overall amount of allocated resources is large enough to satisfy most customers' needs. Due to the highest amount of allocated resources and the highest amount of utilized resources, this model is good for servers which are underutilized.

As can be seen from Fig. 1c, model 3 is a combination of model 1 and model 2. The first function is the shrunk version of model 1 and the second one is the shrunk version of model 2. The model combines the characteristics of both models and therefore eliminates some of the weaknesses. From Table 1, we can see that the overall amount of allocated resources is between model 1 and model 2. This means that it could be good for moderately utilized servers. As shown in Fig. 1c, for customers who have x around 0.5, there is not much difference when x changes. This model could be suitable for new customers because they have a more fluctuating behaviors than existing customers and broker does not have enough records on them to make good resource estimation. This model could let them have stable services and the providers would not waste resources. This

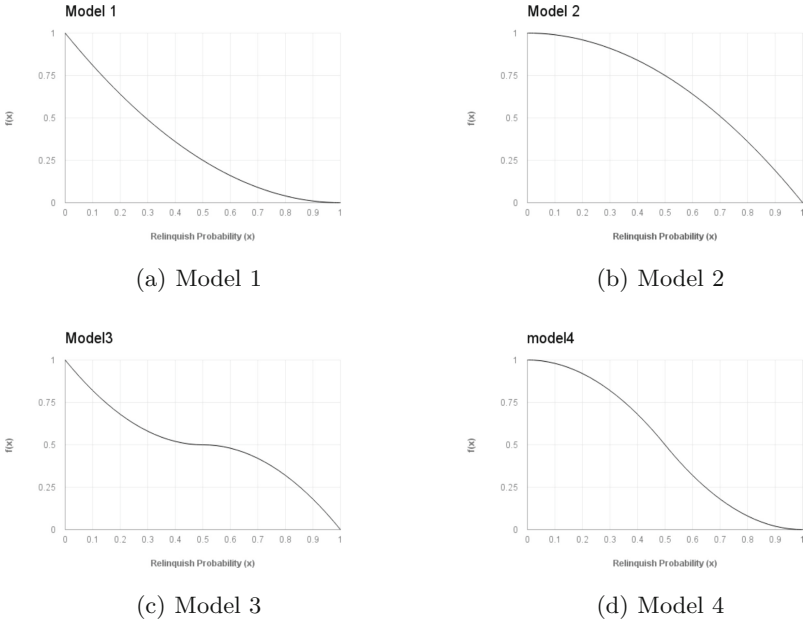


Fig. 1. Relation between relinquish probability and $f(x)$

model could also fit situations where services are expensive and requested by disloyal customers.

Finally, model 4 combines model 1 and model 2 in the opposite way. Similar to model 3, the overall amount of allocated resources is between model 1 and model 2. However, when x is around 0.5, $f(x)$ changes significantly with respect to x . This situation would be a good incentive for users in the middle. In this model, loyal users get much better services than disloyal users. In this case, this model could be a good fit for dealing with loyal users and inexpensive requests by disloyal users in moderately utilized servers.

In this section, several models were proposed and analyzed. These mathematical models will make the resource estimation more realistic and concrete. The customers request is more specific and some parameters in the resource estimation model are removed or modified to make resource estimation more accurate. Each new model has its own pros and cons and therefore, they could fit different situations depending on the current server utilization, type of request, etc.

4 Simulation and Analysis of Results

In this section, we present simulation results conducted with CloudSim 3.0.3 and compare the results with the mathematical results presented in the previous section. For the simulations, we used a PC running Ubuntu 12.04 LTS with 4 GB of RAM.

4.1 The User’s Behavior

To add randomness in the behavior of customers, the Gaussian distribution is used. For example, to simulate a disloyal customer with an average relinquish probability of 0.7, the following Gaussian distribution is used: $X \sim N(0.7, 0.3)$. According to the definition of the Gaussian distribution, 70% of the instances will tend to have a value one standard deviation on either side of the average. In other words, 70% of the values generated from the distribution model will be between 0.7 ± 0.3 . 95% of the values will be between 0.7 ± 0.6 and more than 99% of the values will be between 0.7 ± 0.9 . Since the relinquish probability can only be between 0 and 1, a filter was implemented to discard values lower than 0 and larger than 1.

Table 2. Input parameters setup

Parameters	value
Requested CPU (GHz)	300
Requested memory (GB)	300
Requested storage (GB)	300
Duration (h)	200

4.2 Results

Table 2 lists all the parameters used for the simulation. There will be eleven groups of customers with different average relinquish probabilities ranging from 0 to 1. They will be asking for same services from the same cloud service provider. Each group has ten customers and they will relinquish services on probabilities generated by the Gaussian distribution explained earlier. The mean of the Gaussian distribution is the average relinquish probability of each customer and the standard deviation is 0.3. It is worth noting that all customers are existing customers because that will avoid the unnecessary fluctuation caused by first time customers (since there is no existing record). During the simulation, some important outputs are collected such as the amount of allocated resources, server utilization, etc. In order to calculate the server utilization, it is assumed that the cloud provider has a maximum capacity of 300 gigahertz of CPU, 300 gigabytes of RAM and memory, which is same as the request of each customer. This means that if the customer gets the service he wants, the server utilization will be 100%, which is the same as the mathematical analysis.

From Fig. 2, we can see that the results from the simulations are similar to the mathematical analysis. However, we can notice a slight difference on each end of the curves. This is due to the fact that the standard deviation of the Gaussian distribution is set to 0.3 and we are filtering out some values (all values <0 or >1). For the other parts of the graphs, the simulation results are close to

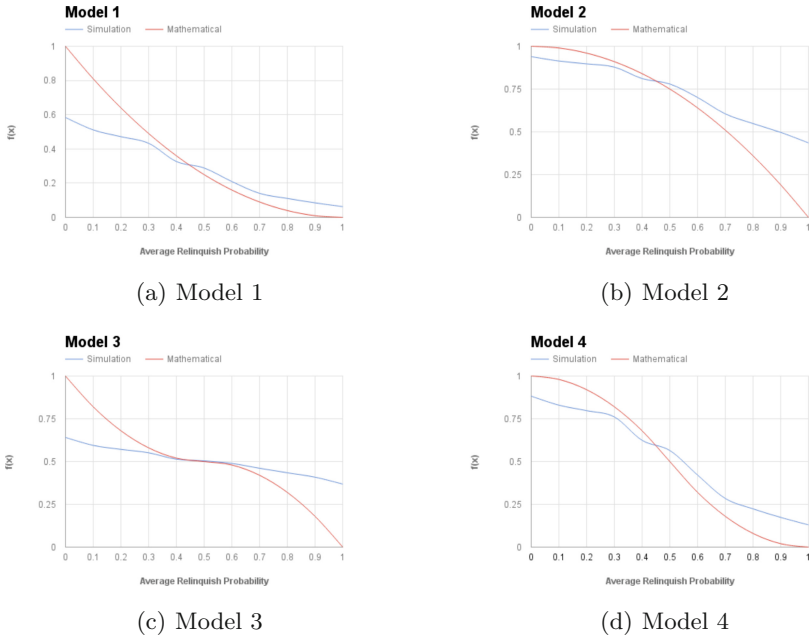


Fig. 2. Comparison between the mathematical model and the simulation

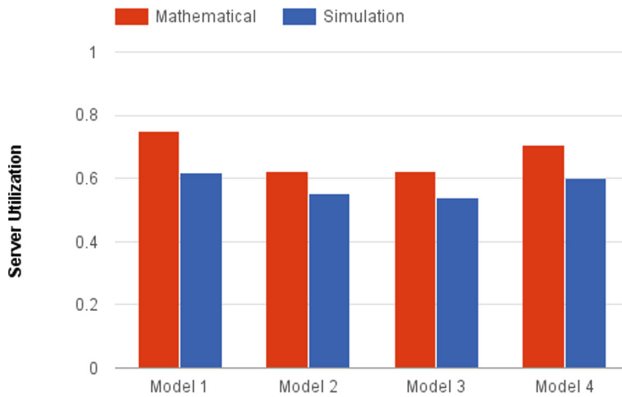


Fig. 3. Server utilization of the proposed models

the mathematical models. In summary, it can be concluded that the simulation results match the mathematical analysis.

To make the comparison more complete, the sever utilization is also regarded as a metric for comparison between the simulation results and the mathematical analysis. Figure 3 shows the server utilization calculated from the mathematical models and the simulation results. The server utilization is computed by dividing the average amount of allocated resources by the maximum amount of resources

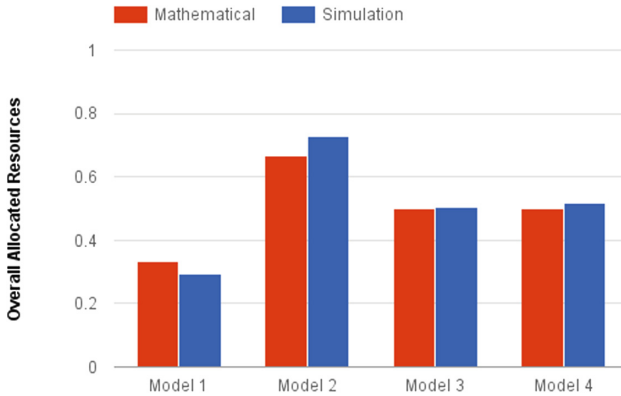


Fig. 4. Overall allocated resources of the proposed models

(i.e. 300). As we can see from Fig. 3, the server utilization of the mathematical models is similar to the server utilization calculated from the simulation results. The small difference is mainly because of the fluctuating relinquish probabilities caused by Gaussian distribution. In conclusion, the proposed models perform the same way as mathematical models as for server utilization. As for the amount of allocated resources, Fig. 4 shows that there is not much differences between the mathematical results and the simulation results. However, one important observation is that each model is assigning a different amount of resources. This justifies the use of different models in different situations. For example, if a cloud provider is underutilized, it might be better to use a model that assigns more resources to increase utilization.

5 Conclusions

In this paper, the model proposed by Aazam and Huh [1] is extended to improve the performance of the broker-based resource estimation model. Instead of using the same scheme for assigning resources for all server loads, we proposed four models implementing different strategies for aligning resources and provide better resource allocation. The parameters from the resource estimation model proposed in [1] have been modified to make the model more realistic. For each model, the pros and cons were analyzed mathematically and the performance of all models was compared under several criteria such as server utilization, the amount of allocated resources, customers' relinquish ratio, and type of service. The comparison between the simulation results and the mathematical analysis shows that the models perform well under simulations, as expected.

As future work, we want to build an adaptive model that will include the four proposed models as modules. Given the conditions upon a request arrival, the appropriate module will be invoked to estimate how much resources should be allocated. Furthermore, we may add more features in the revenue system to make the whole model more comprehensive.

References

1. Aazam, M., Huh, E.: Broker as a service (BaaS) pricing and resource estimation model. In: IEEE 6th International Conference on Cloud Computing Technology and Science (2014)
2. Miller, R.: Google's energy story: high efficiency, huge scale
3. Whitney, J., Delforge, P.: Data center efficiency assessment. Natural Resources Defense Council (2014)
4. Moreno, I., Garraghan, P., Townend, P., Xu, J.: An approach for characterizing workloads in google cloud to derive realistic resource utilization models. In: 2013 IEEE 7th International Symposium on Service Oriented System Engineering (SOSE), pp. 49–60 (2013)
5. Beloglazov, A., Buyya, R.: Energy efficient resource management in virtualized cloud data centers. In: 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid), pp. 826–831 (2010)
6. Wang, C., Yang, C.: A prediction based energy conserving resources allocation scheme for cloud computing. In: IEEE International Conference on Granular Computing (2014)
7. Dabbagh, M., Hamdaoui, B., Guizani, M., Rayes, A.: Energy-efficient cloud resource management. In: 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 386–391 (2014)
8. Biswas, A., Majumdar, S., Nandy, B., El-Haraki, A.: Automatic resource provisioning: a machine learning based proactive approach. In: 2014 IEEE 6th International Conference on Cloud Computing Technology and Science (CloudCom), pp. 168–173 (2014)
9. Aazam, M., Huh, E.: Advance resource reservation and QoS based refunding in cloud federation. In: Globecom 2014 Workshop - Cloud Computing Systems, Networks, and Applications (2014)
10. Amazon Web Services: How AWS Pricing Works. (2015). Whitepaper