A Workload Prediction-Based Multi-VM Provisioning Mechanism in Cloud Computing

Shengming Li, Ying Wang, Xuesong Qiu, Deyuan Wang, Lijun Wang
State Key Laboratory of Networking and Switching Technology
Beijing University of Posts and Telecommunications
Beijing, China
lishengming@bupt.edu.cn, wangy@bupt.edu.cn, xsqiu@bupt.edu.cn, peterwdy@gmail.com, ljwang@bupt.edu.cn

***Abstract*—With the emerging of cloud computing, more and more enterprise organizations begin to migrate their applications to IaaS, which is a more flexible and cheaper alternative to traditional infrastructures. IaaS providers usually offer customers with resources in the form of VM and charge them in a time-based billing model. Meanwhile customers are allowed to dynamically apply for VM resources. However, highly dynamic workload makes customers difficultly determine how much capacity to provision. Furthermore, it is also a great challenge for customers to determine how to choose a VM provisioning scheme to match workload at a low cost. In this paper, we propose a workload prediction-based multi-VM provisioning mechanism to overcome these challenges, which contains an ARIMA workload predictor with dynamic error compensation (ARIMA-DEC) and a time-based billing aware multi-VM provisioning algorithm (TBAMP). The experimental results show that ARIMA-DEC predictor can obviously reduce SLA default rate and TBAMP algorithm can effectively save rental cost comparing to the existing algorithms.**

Keywords-cloud computing; IaaS; workload prediction; multi-VM provisioning.

I. INTRODUCTION

In the traditional scenario, enterprise organizations build their IT infrastructures in advance according to the average or peak workload requirements. However, if the capacity is planned according to the average requirements, the performance of IT system will not be guaranteed when peak workload occurs. And if the capacity is planned according to the peak requirements, part of the resources will remain idle in most of the time. Besides, the enterprise still needs to maintain IT infrastructures on its own. Fortunately, the appearance of cloud computing has broken these limits.

IaaS is a type of cloud computing service wherein enterprise organizations can outsource their IT infrastructures including storage, processing, networking, and other resources. Enterprise customers can access these resources over the internet in a time-based billing model. IaaS cloud platform usually provides these resources in the form of VM, which are usually rented on a fixed time base (e.g., hourly) to customers. For instance, the price of a small Linux VM instance (1 ECU, 1.7GB RAM, 160GB Disk) of EC2 is 0.08$/hr [1]. Furthermore, IaaS cloud platform allows enterprise customers to dynamically add or remove VMs within a few minutes [2].

If the real workload exceeds the provisioned capacity, there will be a need to add new VM resources. However, the performance of enterprise applications may degrade during the period of resource acquisition. Degraded performance will discourage the users of enterprise applications and then reduce enterprise revenues. Therefore, enterprise customers need to accurately predict workload requirements in the near future in order to apply for new VM resources in advance to guarantee the quality of service. Then, after the future workload is predicted, enterprise customers still need to make a decision on how to choose a VM provisioning scheme to match workload requirements and at the same time save the rental cost as much as possible.

At present, there have been a great many researches on solving these problems. Some linear predictors are introduced to solve the problem of workload prediction [3-7]. And all of these works are based on Mean Squared Error (MSE). However, we observed that the performance of enterprise application will degrade when the prediction result is less than the real workload (under-provisioning). On the other hand, if the prediction result is greater than the real workload (over-provisioning), part of resources will remain idle in a short time. This may lead to more resource cost, but the quality of service could be guaranteed. So enterprise customers usually prefer over-provisioning to under-provisioning. And many VM provisioning algorithms are proposed from different perspectives [3-10]. But all of these researches didn't consider the factor of time-based billing. We noticed that the usage cost of a VM is based on a fixed time base and customers will be charged for the full time base even if resources are only used part of the time base. A detailed example will be described in Section III.A.

The main contribution of this paper is as follows:

1). ARIMA-DEC predictor is proposed to help customers forecast the future workload considering the difference between under-provisioning and over-provisioning;

2). TBAMP algorithm is proposed to help customers choose an appropriate VM provisioning scheme to match the predicted workload considering the factor of time-based billing;

The rest of this paper is organized as follows: Section II analyzes the related work. Section III mainly describes the mechanism of VM provisioning we proposed, involving the inspiring scenario, ARIMA-DEC predictor, multi-VM provisioning model and TBAMP algorithm. In Section IV, the

This work was supported by the National High Technology Research and Development Program of China (2013AA013502), the Funds for Creative Research Groups of China (61121061) and the Fundamental Research Funds for the Central Universities BUPT (2013RC1103)

efficiency of ARIMA-DEC predictor and TBAMP algorithm is verified through experiments. Finally, Section V ends the paper with our concluding remarks.

II. Related Work

A good workload predictor can help customers forecast the future workload so that customers can apply for new VM resources in advance to match the workload requirements. Thus, it can reduce the default rate of Service Level Agreements (SLA) and improve the quality of service. Most of the current works introduced the traditional linear predictor to solve the problem of workload prediction, such as ARMA [4, 5] and ARIMA [3, 6, 7]. All of these works are based on MSE, which consider under-provisioning condition and over-provisioning equally. As is stated above in Section I, enterprise customers actually prefer over-provisioning to under-provisioning. Therefore, it is necessary to distinguish between under-provisioning and over-provisioning when the need to predict workload.

In addition, many VM provisioning algorithms were proposed to help customers choose an appropriate VM provisioning scheme to match workload from different perspectives [3-10]. Some studies focused on single-VM provisioning. For instance, Gueyoung Jung et al. presented a single-VM provisioning algorithm based on performance model they proposed [4], Nilabja Roy et al. proposed a single-VM provisioning algorithm based on queuing network system model [8] and Valeria Cardellini et al. formulated the issue of single-VM provisioning as an optimization problem combined reactive and proactive heuristic policies [9]. However, IaaS providers usually offer enterprise customers with different types of VMs. It can make enterprise customers have more flexible choices to support their applications. Other studies focused on multi-VM provisioning. Zhang Fan et al. proposed a mathematical multi-tier framework for small and large VM allocation [10]. Upendra Sharma et al. proposed Kingfisher, which is a multi-VM provisioning algorithm that provides efficient support for elasticity in the cloud [3, 6]. However, all of these researches didn't consider the factor of time-based billing. In the multi-VM provisioning scenario, it is very common for enterprise customers to release unneeded VMs. Since the usage cost of a VM is based on a fixed time base, it may cause a cost waste to customer when a VM is released before a time base completes. Hence, it is necessary to consider the factor of time-based billing when there is a need to choose a new VM provisioning scheme to match the workload requirements.

III. Multi-VM Provisioning Mechanism

A. Inspiring Scenario

We assume such a scenario: IaaS providers offer customers with two types of VM (Small and Medium). The price of the Small instance (S) is $0.08/hr, and the requests processing rate of the Small instance (S) is 100req/min. The price of the Medium instance (M) is $0.16/hr, and the requests processing rate of the Medium instance (M) is 250req/min. At the 0^{th} minute, the workload is 150req/min and two S-type VMs are applied to handle requests. After 70 minutes, the workload jumps from 150 to 250req/min.

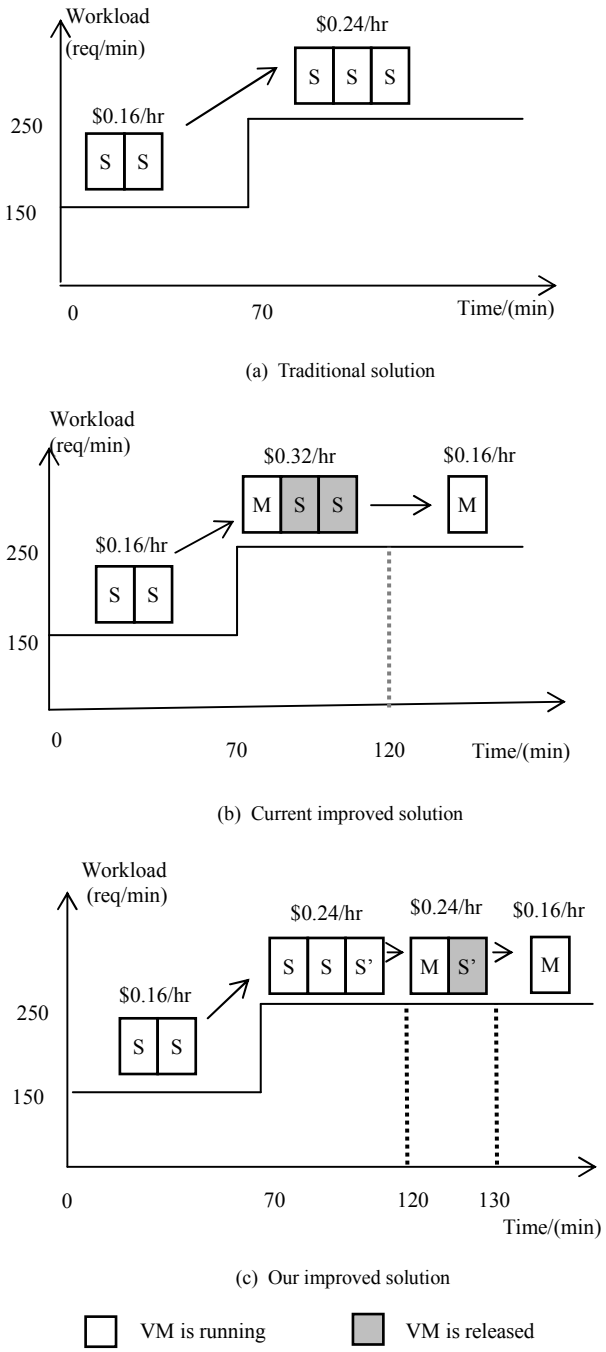

Figure 1. VM provisioning of three solutions

In such a scenario, different VM provisioning strategies will result in different provisioning schemes as shown Figure1. When the provisioned capacity of current VM scheme (2S) are not sufficient to meet the application requirements at the 70^{th} minute, the traditional solution (Cost-Oblivious) [4, 8, 9] will retain all of the current VM resources (2S) and add appropriate number of VMs (1S) to meet residual workload as shown Figure1 (a). However, it is obvious that 1M-scheme is better than 3S-scheme to meet the workload of 250req/min. [3, 6] proposed an improved solution (Kingfisher cost-aware). The algorithm selected the "optimal" VM provisioning scheme (1M)

according to total workload as shown in Figure1 (b). However, the factor of the time-based billing is not considered. The IaaS cloud platform use a time-based billing that allows VMs to be rented on a fixed time base (e.g., hourly). Although the two S-type VMs have been released, the customer still needs to pay for the unused time (50 minutes).

If an S-type VM (S') is added at the 70th minute, then an M-type VM is added and other VMs are released at the 120th minute, the customer will pay less money as shown in Figure1 (c). From this inspiring scenario, it is necessary to consider the factor of time-based billing when there is a need to choose a new VM provisioning scheme to match workload at a low cost.

B. ARIMA-DEC Predictor

Before choosing an appropriate VM provisioning scheme to match workload, enterprise customers should know firstly how much capacity to provision. So the problem of workload prediction needs to be solved.

If customers want to add new VM resources, there is a need to make a call on the IaaS Cloud API. And the VMs should be booted up with the specified images. It usually takes several minutes to complete this process. So if the customer begin to apply for new VM resources just when the real workload is higher than the provisioned capacity, it will probably lead to performance degradation of enterprise applications during the period of resource acquisition. Degraded performance will discourage customers and affect revenue. Over-provisioning may make enterprise customers pay a bit more, but the effects of under-provisioning will increase SLA default rate. Enterprise customers usually prefer over-provisioning to under-provisioning. Therefore, a good workload predictor should not only forecast the future workload, but also distinguish between under-provisioning and over-provisioning.

In this paper, the idea of linear predicting with dynamic error compensation is used to solve the problem. Facing the condition of under-provisioning, the prediction error of the previous step is used to compensate the current step of forecasting. And no compensation will be made in the condition of over-provisioning.

The ARIMA-DEC predictor consists of two parts: the ARIMA prediction part and the error compensation part. The ARIMA prediction part could be denoted as AP_tX_{t+1}, where X_t is the real workload at time t and AP_tX_{t+1} predicts the future value X_{t+1} of a time series according to its historical data using ARIMA predictor. And the error compensation part could be denoted as $f_{ec}(X_t\text{-}AP_{t-1}X_t)$, where f_{ec} is a function to calculate the error compensation from the prediction error of the previous step. Combining them together, the ARIMA-DEC predictor could be denoted as ADP_tX_{t+1}, which can be written as (1).

$$ADP_tX_{t+1} = AP_tX_{t+1} + f_{ec}(X_t - AP_{t-1}X_t) \quad (1)$$

A direct way of implementing f_{ec} is to let it exactly equal to the prediction error of the previous step, i.e. $f_{ec}(x) = x$. In order to prevent the over compensation, α is introduced as the max value of the compensation. As mentioned in the previous analysis, we only compensate for the condition of under-provisioning as shown (2).

$$f_{ec}(x) = \min(\max(x, 0), \alpha) \quad (2)$$

The ARIMA-DEC algorithm proposed in this paper is described as Algorithm 1. The time series zt is calculated according to the history series yt and the differential transform d in line 2. If zt isn't a stationary sequence, it should increase d and recalculate time series zt in line 3. The orders of model can be calculated using the ACF/PACF method or AIC/BIC information criterion in line 4. And the parameters of model can be estimated using moment estimation or maximum likelihood estimation in line 5. In line 6, the final prediction result is the sum of the result of ARIMA predictor part AP_tX_{t+1} and the result of error compensation part $f_{ec}(X_t\text{-}AP_{t-1}X_t)$.

Algorithm 1: ARIMA-DEC Algorithm

1. Get the history series $X1, X2...Xt$, set the order of differential transform, d=0
2. calculate time series zt, zt=∇dXt
3. Check time series zt, if it is a stationary series, go to 4 else, $d = d + 1$, go to 2
4. Estimate order of model using the ACF/PACF method or AIC/BIC information criterion
5. Estimate parameters of model (AP_tX_{t+1}) using the moment estimation or maximum likelihood estimation
6. $ADP_tX_{t+1} = AP_tX_{t+1} + f_{ec}(X_t - AP_{t-1}X_t)$

C. Multi-VM Provisioning Model

After solving the problem of workload prediction, enterprise customers still need to make a decision on how to choose VM provisioning scheme to match workload requirements and at the same time save the rental cost as much as possible. In this section, a more efficient VM provisioning model is proposed, which can be stated using the following integer linear program (ILP) as shown in (3)-(6).

$$\min\ Cost = \sum_{i=1}^{N} X_i P_i + \mu \sum_{i=1}^{N} \sum_{j=1}^{A_i} PT_{ij} E_{ij} \quad (3)$$

subject to the constraints

$$\sum_{i=1}^{N} X_i C_i \geq \beta \quad (4)$$

$$Eij = \begin{cases} 1, & X_i < A_i \ and\ j \in [1, A_i - X_i] \\ 0, & others \end{cases} \quad (5)$$

$$Xi \in \{0, 1, 2, ...\}, \forall i \in \{1, 2, ...N\} \quad (6)$$

Equation (3) is the objective of the problem, which not only considers the cost of adjusted VM scheme but also takes the cost of released VMs into account due to the effect of the factor of time-based billing. Let N represents the number of VM types supported by the IaaS platform; let X_i represents the number of type-i VM in the adjusted VM scheme; let P_i represents the price of type-i VM; let A_i represents the number

of type-i VM in the current VM scheme; let T_{ij} represents the unused time of the j^{th} VM of the A_i type-i VMs if the VM is released now ($T_{i1} \le \cdots \le T_{ij} \le \cdots \le T_{iA_i}$); E_{ij} is an integer variable that can take values of 0 or 1, and value 1 indicates that the j^{th} VM of all type-i VMs will be released while value 0 indicates that the VM will be remained; $\sum_{i=1}^{N} X_i * P_i$ represents the cost of adjusted VM scheme and $\sum_{i=1}^{N}\sum_{j=1}^{A_i} P_i * E_{ij} * T_{ij}$ represents the cost of VMs which are released. μ is a constant and its value will be discussed in section IV.B.2).

The constraint in equation (4) is used to ensure that the provisioned capacity is greater than the predicted workload. Let C_i represents the provisioned capacity of type-i VM; let β represents the predicted workload. Equation (5) is used to determine the j^{th} VM of all type-i VMs whether it will be released and Equation (6) is the basic constraint.

Considering the factor of time-based billing, in order to reduce the delay of applying for a new VM, we propose to maintain a set of released VMs. The VMs in the set are still controlled by the customer. And a released VM in the set will be removed from the set when the unused time of this VM is below a threshold. When a new VM is needed, it is convenient to get appropriate VM from the VM set.

D. TBAMP Algorithm

The ILP is NP-hard problem. Indeed, there is no efficient method to solve the problem in polynomial time. Currently, brute-force method, recursive backtracking method and brand bounding method are usually used to solve the problem.

Among them, the efficiency of brute-force method is too low. Although the brand bounding method is generally used, there is not a simple way to find branch node due to the nonlinear relationship between E_{ij} and X_i. Therefore, we select recursive backtracking method to solve the problem. The TBAMP algorithm is described as Algorithm 2.

Algorithm2: TBAMP Algorithm

```
TBAMP(K)
1  if K=N
2      then X_N ← border(K)
3       cost ← Σ_{i=1}^{N} X_i P_i + μ Σ_{i=1}^{N} Σ_{j=1}^{A_i} P_i T_ij E_ij
4       if cost<minCost
5            then minCost ← cost and  record the VM
                  scheme as current optimal solution
6  else bound(K)
7  M ← border(K)
8  for i ← 0 to M then
9     do  X_K ← i
10         TBAMP(K+1)
```

Line 1 to line 5 is the export of our recursive algorithm. If the cost of part of the VM provisioning scheme is greater than the cost of the current optimal solution, *bound(K)* can be used to cut the branch off (as seen in line 6). Knowing the number of VM of type-1, type-2, ..., type-(k-1) , *border(K)* is used to calculate the number of type-k VM only using the VM to meet the remaining workload (as seen in line 7). *border(K)* can be used to dynamically construct solution space to avoid a lot of unnecessary search. For space reason, the expression of *bound(K)* and *border(K)* are not explicitly described in Algorithm 2. The optimal VM provisioning scheme is found by recursive backtracking from line 8 to line 10.

The time complexity of Cost-Oblivious, Kingfisher and TBAMP is $O(n^N)$. Brand bounding method (Cost-Oblivious, Kingfisher) improves search efficiency by pruning. If the number of VM type is large, the efficiency of our proposed algorithm is not good. However, we observed that the number of VM type for the specific application class is usually less than 4. For example, Amazon EC2 provides customers with 4 kinds of Standard Instance, 3 kinds of High-Memory Instance, 2 kinds of High-CPU Instance, 3 kinds of Cluster Compute Instance and so on. Thus, the time complexity of TBAMP algorithms can be accepted.

IV. EXPERIMENTAL EVALUATION

A. Inputs of Simulation

The workload of web application changes a lot during a day. For instance, Figure 2 depicts a real-world workload scenario of the FIFA 1998 soccer world cup website [11]. We analyzed these data from May 1 to May 10. The user requests are counted every 15 minutes. The data from May 1 to May 7 are used to calculate related parameters of workload prediction model and the other data are used to test the efficiency of the model.

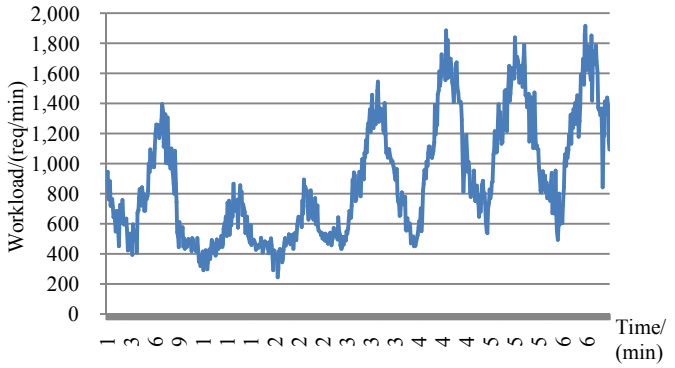

Figure 2. Workload change from May 1 to May 7

In addition, a virtualized IaaS cloud platform is also need to be simulated. The platform is assumed to support three types of instance-classes—small, medium and large instances. The price is set according to EC2 [1] and the processing capacity of different VM instance can be got by an empirical approach, which estimates the processing capacity by running the enterprise application on different VM instance in the real scenario [6], detailed information described in TABEL I. Furthermore, the length of time base is set 1 hour by default.

TABLE I. PRICE AND CAPACITY OF DIFFERENT VM INSTANCE

VM Type	Price($/hr)	Capacity(req/min)
Small	0.08	60
Medium	0.16	140
Large	0.32	300

B. Experiment Result and Analysis

1) workload predictor

A 3-day simulation is conducted based on the simulation scenario formulated in section IV.A. Two predictors are compared in our experiment and related experiment results are described in Table II. When the current workload is greater than the provisioned capacity, it is considered as a SLA default.

TABLE II. INFORMATION COMPARED BETWEEN ARIMA AND ARIMA-DEC

Predictor	ARIMA	ARIMA-DEC
MSE(req/min)	94.5	107.6
COST($)	72.59	75.46
SLA Default Rate	37.84%	13.19%

Table II shows that ARIMA has lower MSE and cost, but its SLA default rate is relatively high. Without performance assurances, it will discourage the users of enterprise applications and then reduce enterprise revenues. And the experimental results show that ARIMA-DEC can obviously reduce SLA default rate with a little more cost.

2) The Analysis of μ

The influence on TBAMP with different value of μ is analyzed through experiments, which is illustrated in Figure 3.

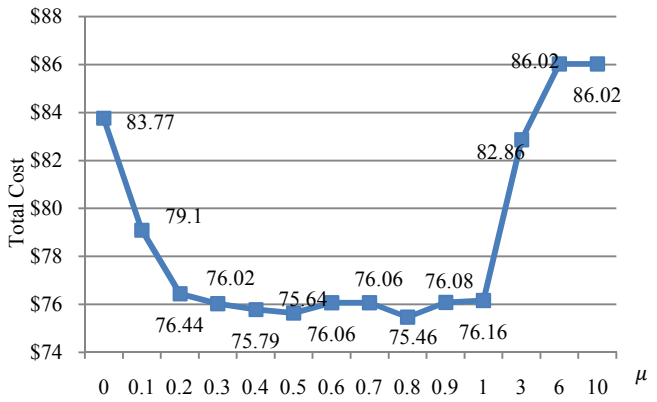

Figure 3. The influence on total cost with different value of μ

The total cost is the expenses that customers need to pay from May 8 to May 10. Figure3 shows that the total cost is maintained at a low level when the value of μ is between 0.2 and 1. When the value of μ is 0.8, the lowest cost is got. We set μ to 0.8 in later experiments.

3) The Analysis of the Factor of Time Base

The IaaS cloud platform usually charges customers based on the time-based billing, which allows VMs to be rented on a fixed time base. We want to observe the influence on algorithm with different length of time base (LTB). The length of the time base is set in the experiment to 0, 20, 40, 60, 80, 100, 120 minutes in turn and the results are presented in Figure4.

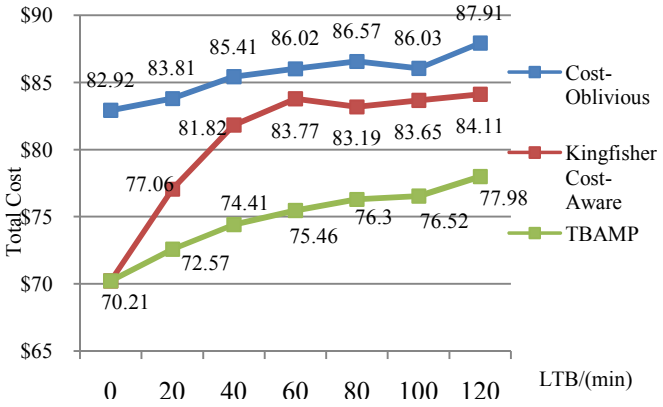

Figure 4. The influence on total cost with different lengh of time base

From the experimental results, the following conclusions could be drawn: 1) When the length of the time base is 0, Kingfisher Cost-Aware and TBAMP have the same cost; 2) With the increase of the length of the time base, the cost of Kingfisher Cost-aware is obviously more than the cost of TBAMP; 3) The cost of Cost-Oblivious maintains a high level.

If the length of the time base is 0, IaaS providers instantly stop charging when the customer releases a VM. So the provisioning scheme of TBAMP is the same with Kingfisher Cost-Aware. When the length of the time base becomes larger, TBAMP can save money by considering the factor of time-based billing. Since Cost-Oblivious adopts the idea of retaining the current VM resources, the number of VMs which are released is little when VM provisioning scheme need to be adjusted. So Cost-Oblivious is affected less by the factor of time-based billing.

Nowadays, the main IaaS cloud platform (such as Amazon EC2 [1] and windows Azure [12]) set the time base to 60 minutes. If the time base is too small, enterprise customers may frequently add or remove VMs to minimize their cost and it will increase the management difficulty of IaaS cloud platform. So the time-based billing will exist for a long time. Therefore, the proposed algorithm is very useful for helping customers to reduce rental cost in such a scenario.

4) The Analysis of the efficiency of TBAMP Algorithm

In order to verify the efficiency of TBAMP algorithm, we compared the cost of different algorithms according to valid cost, invalid cost and total cost. The valid cost stands for the rental cost that the customers need to pay for the VM that actually be used. And the invalid cost stands for the rental cost that the customers need to pay for the unused time of the VMs that have been released. And the total cost is the sum of valid cost and invalid cost. The experimental results are presented in Figure 5.

The experiment results are the same with our previous expectation (see in Section III.A). In the aspect of valid cost, Kingfisher Cost-Aware can save the most money. In the aspect of invalid cost, Cost-Oblivious can save the most money. In the aspect of total cost, TBAMP can save the most money. TBAMP algorithm can effectively balance the valid cost and invalid cost by considering the factor of time-based billing. In

detail, TBAMP can save 12.3% compared with Cost-Oblivious and save 9.9% compared with Kingfisher Cost-Aware.

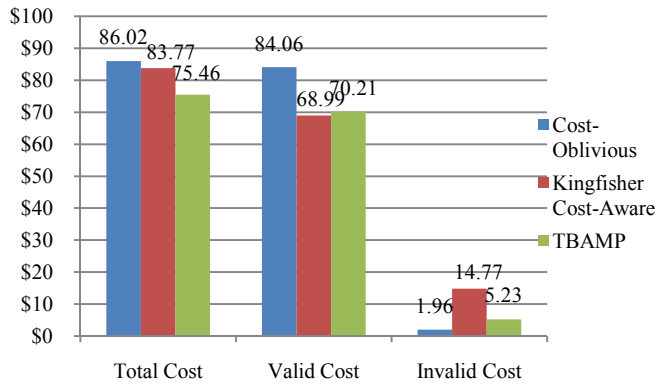

Figure 5. The analysis of the efficiency of different algorithms

V. CONCLUSION

In this paper, we propose the ARIMA-DEC workload predictor as well as the TBAMP algorithm. ARIMA-DEC workload predictor adopts the idea of ARIMA model combined with dynamic error compensation to solve the problem of workload prediction. The experiment results show that ARIMA-DEC predictor can obviously reduce SLA default rate. And TBAMP algorithm not only considers the cost of adjusted VM scheme, but also takes the cost of released VMs into account due to the effect of the factor of time-based billing. The experimental results show that TBAMP algorithm can effectively save rental cost comparing to the existing algorithms.

REFERENCES

[1] "Amazon Elastic Compute Cloud (EC2) ,"[Online]. Available: http://aws.amazon.com/ec2 [Accessed Aug. 20, 2012]

[2] M. Armbrust, A. Fox, et al, "Above the Clouds: A Berkeley View of Cloud Computing," U.C. Berkeley, Tech. Rep. UCB/EECS-2009-28, Feb 2009.

[3] U. Sharma, P. Shenoy, S. Sahu and A. Shaikh, "Kingfisher: Cost-aware Elasticity in the Cloud," in International Conference on INFOCOM, Shanghai, China, pp.206-210, April 2011.

[4] N. Roy, A. Dubey and A. Gokhale, "Efficient Autoscaling in the Cloud using Predictive Models for Workload Forecasting," in International Conference on Cloud Computing(CLOUD), Washington, DC, USA, pp.500-507, July 2011.

[5] G. Jung, M. A. Hiltunen, K. R. Joshi, R. D. Schlichting and C. Pu, Mistral: Dynamically Managing Power, Performance, and Adaptation Cost in Cloud Infrastructures, in International Conference on Distributed Computing Systems(ICDCS) , Genova,Italy, pp.62-73, June 2010

[6] Upendra S, Prashant S and Sambit S. "A Cost-Aware Elasticity Provisioning System for the Cloud," in International Conference on Distributed Computing Systems (ICDCS), Minneapolis, MN, USA, pp.559 – 570, June 2011.

[7] W. Fang, Z. H. Lu, J. Wu and Z. Y. Cao. "RPPS: A Novel Resource Prediction and Provisioning Scheme in Cloud Data Center," in International Conference on Services Computing (SCC), Honolulu, HI, pp. 609 – 616, June 2012.

[8] R. N. Calheiros, R. Ranjan, and R. Buyya. "Virtual Machine Provisioning Based on Analytical Performance and QoS in Cloud Computing Environments," in International Conference on Parallel Processing (ICPP), Taipei, China, pp.295-304, Sep. 2011.

[9] V. Cardellini, E. Casalicchio, F. L. Presti and L. Silvestri. "SLA-aware Resource Management for Application Service Providers in the Cloud," in International Symposium on Network Cloud Computing and Applications (NCCA), Toulouse ,France, pp.20–27, Nov. 2011 .

[10] F. Zhang, J. W. Cao, H. Cai, J. J. Mulcahy and C. Wu. "Adaptive Virtual Machine Provisioning in Elastic Multi-tier Cloud Platforms," in International Conference on Networking, Architecture and Storage (NAS), Dalian, Liaoning ,China, pp.11-19, July 2011.

[11] "Dataset consists of all the requests made to the 1998 World Cup Web site between April 30, 1998 and July 26, 1998," [Online]. Available: http://ita.ee.lbl.gov/html/contrib/WorldCup.html [Accessed June 2, 2012]

[12] "Windows Azure pricing model," [Online]. Available: https://www.windowsazure.com/[Accessed Aug. 20, 2012]