International Journal of Grid and Distributed Computing
Vol. 9, No. 9 (2016), pp.25-36
http://dx.doi.org/10.14257/ijgdc.2016.9.9.03

A Differentiated Virtual Resource Allocation Strategy Based on Game Model for Multi-tenant Applications

Chen Ningjiang, Tan Ying, Li Xiang, Liang Xiaoyu and Huang Ruwei

School of Computer Science and Electronic Information,
Guangxi University, Nanning, 530004
chnj@gxu.edu.cn; tanying90@foxmail.com; lixianggxu@163.com;
lxh9408@163.com; ruweih@126.com

Abstract

Resource allocation for multi-tenancy application is one key issue for cloud service providers. To deal with the competitive problem of limited resources among tenants, this paper proposes a resource allocation strategy based on game model, which combines tenants' SLA and their resource utilization, in order to deal with the shortcomings of the traditional allocation strategies that consider few about differentiated resource requirements. A framework for multi-tenant resources allocation based on game is presented, and the corresponding game model is established. The related processes auction and resource allocation are designed. The experimental results indicate that it can meet the personalized and differentiated demand of multi-tenant better, and help to improve the system's overall resource utilization and decrease the operating costs of cloud service providers.

Keywords*: multi-tenant; resource allocation; game model; SLA; cloud computing*

1. Introduction

In order to improve the quality of service (QoS), cloud service providers (like Amazon, Azure, etc.) and the tenants applying for resources are usually required to sign specific Service Level Agreement (SLA) to regularize the behaviors and obligations of both parties. Multiple tenants apply resources from cloud service providers. However, the resources of the cloud service providers are limited, which may lead to a battle for resources among tenants. If the resource requested by the tenants cannot be met, there are decreased performance, service failure and violation of tenant's SLA. Further, decreased resource utilization and high SLA violations will greatly increase the cloud service providers' operating cost. There have been a number of works that introduce economics methods into the field of resource allocation in order to achieve better management. Online game strategy is a typical application of economics. It provides the support for computing resource allocation by means of solving the problem of resource game between the auctioneer (cloud service providers) and the bidders (tenants).

Traditional researches like [1] and [2] designed certain resource management mechanism based on users' SLAs for virtual machines (VM), but they didn't effectively cope with the fierce resource competition. Michael *et al* [3] established a workload model and a punishment model to monitor the SLAs and deal with resource competition, without considering the priority of the task. Rajkamal *et al* [4] put forward a resource allocation algorithm based on priority, which determines the specific resource configuration for each request based on the user's priority so as to guarantee high-priority users, while it may reduce the resource utilization of the overall system. Guiyi *et al* [5] used the game theory to optimize the allocation results in a specific cloud computing framework. Niyato *et al* [6] analyzed resource profits and shared allocation problems by using levels of cooperative

ISSN: 2005-4262 IJGDC

games and stochastic linear programming games. Wei *et al* [7] also studied the allocation problem of task resources by game theory. Though the works in [5-7] used game theory to solve resource allocation in cloud applications, they ignored the awareness of resource utilization at runtime.

The allocation of resources should consider comprehensively the tenant-defined SLA and the dynamic resource utilization in the system. A dynamic resource allocation strategy for data center was researched in [9] and a multifactor-awarded performance prediction strategy for cloud service was implemented in [10] in order to reduce the destruction of SLA. Zheng *et al* [11] proposed a resource allocation model based on three-party game among users, virtual resources and resource providers, which ensured the benefits of the resource providers as well as optimized the utilization and the users' fees based on wholesale price leverage. They mainly perform resource scheduling by creation or migration of virtual machines. However, when virtual machines are coarse-grained created or deleted, it is likely to cause greater extra overhead and reduce overall resource utilization. In addition, the performance bottleneck is often caused by the finer factors such as CPU, memory, and I/O. In this paper, a fine-grained resource allocation strategy based on game model is proposed. On the basis of the differentiated tenant's SLA, with the resource forecasting based on Kalman filter [12], the resources are allocated online based on game processing.

2. The Game Model for Resource Allocation

We design a multi-tenant auction framework based on game model as the work foundation, as shown in Figure 1. For the two major game participants, cloud service providers and tenants, the framework is divided into three parts: multi-tenant application deployment, resource monitoring and resource allocation.

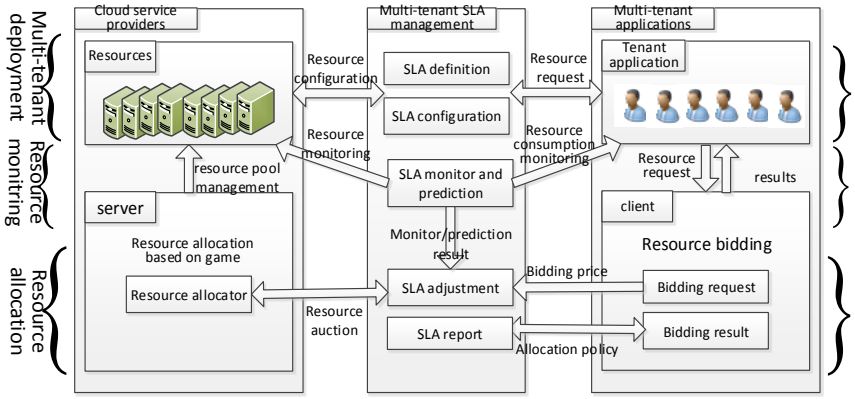


Figure 1. The Framework of Multi-Tenant Resources Allocation based on Game

Multi-tenant applications are deployed in virtual machines provided by cloud service providers. The tenants apply the resources and sign the SLAs from the service providers. The cloud service providers assign on-demand resources for multi-tenant applications from the resource pools. At runtime, the resource consumptions of tenants are monitored, including CPU utilization, average response time, average throughput, etc. Accurate resource monitoring helps to get effective resource demand forecast, and provides the decision-making basis for the game. The general principle of resource allocation is to meet the SLAs of tenants and maximize the resource utilization of cloud service providers. There is bound to be competitive among multiple tenants.

This paper focuses on the implementation of resource allocation strategy based on game model. The tenants and the cloud service providers are the game participants. The

tenants, as the bidders, submit their own bid price, bidding policy and expected profit to the service providers, based on the forecast of resource needs. Cloud service providers determine the final allocation policy based on the SLA and the emergency of resources of tenants, to achieve the game equilibrium. After determining the final resource allocation decision, the cloud service providers call the underlying interfaces to execute resource adjustment.

The key to solve the resource conflict for game participants effectively depends on the game rules that are defined as game benefit function. The game participants include multiple tenants and cloud service providers. Tenant i (i = 1, 2, ... , n) as auction participant shares the resource pools on the servers. The changing workload may lead to dynamic changes of resource demand of the tenants. When the workload is too heavy, the pressure put on the tenant may cause the SLAs are unsatisfied any more. Therefore, tenant i sends auction request B_i= [p_i, q_i] to apply for a certain amount of extra resources from resource pool. Here, p_i refers to the price that tenant is willing to pay, while q_i represents the resource amount that the tenant wants to get. The cloud service providers are regarded as auctioneers, collecting the information about each tenant's bid B = [B_1, B_2, ... , B_n] and the SLAs of the tenants, calculating the usage of resources, and determining the resource allocation.

In the actual bid process, the price p_i provided by auctioneer is positively correlated the wanted resource quantity (a_i). In our model, assuming that they have some kind of linear relationship, for simplification, there has a_i (p_i) = p_i. In the traditional bid, with the consideration of fairness, if tenant i bids success, the resources are assigned for it according to the following allocation rule, as shown as Formula (1).

$$a_i(p_i) = \begin{cases} \min(q_i, x_i Q), \sum_{i=1}^{N} q_i \le Q \\ x_i Q, \sum_{i=1}^{N} q_i > Q \end{cases} \quad (1)$$

Where $x_i = {}^{p_i}/_{\sum_{i=1}^{N} p_i}$, $x_i \in [0,1]$, q_i is the amount of resources that tenant i wants, and Q is the total resource.

When tenant i participants the auction with B_i= [p_i,q_i], the resources are allocated based on the tenant's demand, if the resources provided by the auctioneers is enough. But when the resources cannot satisfy the needs of all tenants, the traditional auction adopts fair allocation strategy, which assigns the appropriate resources to each tenant according to the proportion of resources applied to the total resources. However, this strategy is not suitable to resource allocation for multi-tenant applications. As for the tenants with high SLA, fair allocation strategy may reduce their needed resources and cannot satisfy their SLAs, so that the violation of SLA increases and the trust of service provider decreases. For the tenants with ordinary SLA but extremely urgent resource requirement, the resources obtained are not enough, and the tenants' minimum requirements can't be met. Therefore, we propose a multi-tenant resource auction rule based on SLA, as shown as Formula (2).

$$a_i(p_i) = \begin{cases} q_i, \sum_{i=1}^{N} q_i \le Q \\ G_i Q, \sum_{i=1}^{N} q_i > Q \end{cases} \quad (2)$$

where $G_i \in [0,1]$ is the game allocation factor.

In our strategy, what is different from the traditional strategy is that the game allocation factor G_i is introduced. The expected resources of tenant are related to the SLA and the urgent degree of resource of tenant intimately. The game allocation factor can be solved as follows.

(1) According to the SLAs of tenants, the tenants are divided into two groups including high-SLA tenants VS= [vs_1, vs_2,…, vs_l] and normal-SLA tenants NS=[ns_1,ns_2,…,ns_m], and there is l+$m \leqslant N$.

(2) According to the resource urgency of tenants, the tenants are divided into three

classes: A, B, C. The resource utilization ranges of the three classes are defined by rule of thumb [13]. The following regulations are given: the resource utilization r_i of class A is located in the range of [0.6, 0.7], r_i of class B is located in (0.7, 0.8], and r_i of class C is (0.8, 1].

(3) Firstly the tenants are divided into A, B, C classes, and then the tenants in each class are sorted according to their SLAs' level again. For the tenants in the same class having the same level SLA, they are further prioritized according to the resource utilization. The total priority list of all tenants is obtained, which is used by the service providers for allocating resource.

(4) If there are the tenant whose resource requirement can't be met, that is G_i=0, then the next round of resource auction will be started; on the contrary, the satisfied tenants will be removed from the list. Supposed the length of the priority list is *t*, there are $\sum_{i=1}^{m-1} q_i \le Q \le \sum_{i=1}^{m} q_i, m < t$, and $G_i = {q_i}/{\sum_{i=1}^{m} q_i}$.

Thus, the introduction of G_i is helpful for meeting the individual resource requirement of tenants than the traditional fair allocation.

In the auction, the tenants need to give a reasonable bid price according to its own load and resource utilization. The higher SLA of tenant is, the higher the load level is, the higher resource utilization is, the tenant needs more resources urgently and may obtain more resources by offering higher price. In the paper, given the bid price of tenants *i* is p_i, we have the price function $p_i = p\ (r_i)$. The resource consumptions of the tenants are monitored, and the future resources that the tenants require will be predicted by using Kalman filter. In order to obtain the total profit of tenant's bidding, the incomes and the costs of the tenant are needed to be analyzed under the bidding vector. The tenant's bidding price p_i reflects the actual load of the tenant, and it is linear with the cost c_i of tenant *i*, as defined as Formula (3).

$$c_i\ (p_i) = p_i \qquad (3)$$

As for tenant *i*, the profit gained under the bidding vector can be defined as Formula (4).

$$u_i = v_i(a_i) - c_i(p_i) \qquad (4),$$

Where $v_i\ (a_i)$ is the utility evaluation function for tenant *i* whose allocated resource is a_i.

The game utility function reflects the performance of the tenants after resource allocation, and it needs to meet certain conditions including: 1) the change of**Error! Reference source not found.**for a_i should have a certain stability; 2)**Error! Reference source not found.**should increase with the increase of a_i, the higher the degree of satisfaction of the SLA is got, and the greater the total benefit is got; 3) when $v_i(a_i)$ increases to a certain extent, it should tend to be stable, since the profit of the tenant is constrained. The common utility evaluation function is as follows:

$$v_i\ (x) = \ln\ (1+q_i x) \qquad (5)$$

This function combines the advantages of the linear relationship function and the natural logarithm function, so it has better convergence and stability.

How to choose the right bidding price and to maximize of the resources allocation is the key to the multi-tenant game decision-making. The tenants should choose the bid price that is able to guarantee the maximum utility function, so the problem of the selection of the bid price is converted to Formular (6):

Error! Reference source not found. (6)

Assuming that A_i is the bidding policy of tenant *i*, A_{-i} is the bidding policies of other tenants besides tenant *i*, we give the following definition.

Definition 1: Let the bidding policy vector is**Error! Reference source not found.**, if the condition **Error! Reference source not found.** is satisfied, then for all *i* and all a_i, **Error! Reference source not found.** is Nash equilibrium [14].

Theorem 1: The Formula (6) is a Nash equilibrium point, and there is:

Error! Reference source not found., and $\gamma = \sum_{i=1}^{k} q_i Q \Big/ \sum_{i=1}^{k} q_i Q + 1 - Q$.

Proof:

According to Formula (1), Formula (3), Formula (4), and Formula (5), formula (6) is solved as the follows.

$\max u_i(p_i) = \max u_i(a_i) = vi(a_i) - ci(p_i) = \ln(1+ q_i a_i) - ci(p_i) = \ln(1+ q_i p_i) - p_i$
$= \ln (1+ q_i G_i Q) - G_i Q;$

And $\sum_{i=1}^{k} G_i \le 1, G_i \in [0,1]$;

Let

$$L(G_i, G_{-i}) = \max \; u_i(p_i) - \gamma\left(\sum_{i=1}^{k} G_i - 1\right)$$
$$= \ln(1 + q_i G_i Q) - G_i Q - \gamma\left(\sum_{i=1}^{k} G_i - 1\right) \quad (7)$$

The partial derivative of L (G_i, G_{-i}) is worked out as follows:

$$\frac{\partial L}{\partial G_i} = \frac{q_i Q}{1 + G_i q_i Q} - Q - \gamma \quad (8)$$

Let **Error! Reference source not found.**, then there is:

$$G_i = \frac{q_i Q - Q - \gamma}{q_i Q(Q + \gamma)} \quad (9)$$

Because **Error! Reference source not found.**=1, from Formula (9) there is Formula (10):

$$\gamma = \frac{\sum_{i=1}^{k} q_i Q}{\sum_{i=1}^{k} q_i Q + 1} - Q \quad (10)$$

Since $a_i = G_i Q$, finally **Error! Reference source not found.**there is:

$$a_i = \frac{q_i Q - Q - \gamma}{q_i (Q + \gamma)} \quad (11)$$

We also can prove easily that based on the utility function satisfying the given conditions, if the decision vector $A = (a_1, a_2 \ldots a_N)$ is the Nash equilibrium point, then this is the only one. Therefore, based on the given utility function in the paper, the presented resource allocation based on game model can get the results that bring the trade-off between the bidding prices and the profits of the tenants, as well as the maximization of the profit.

3. The Resource Allocation Strategy

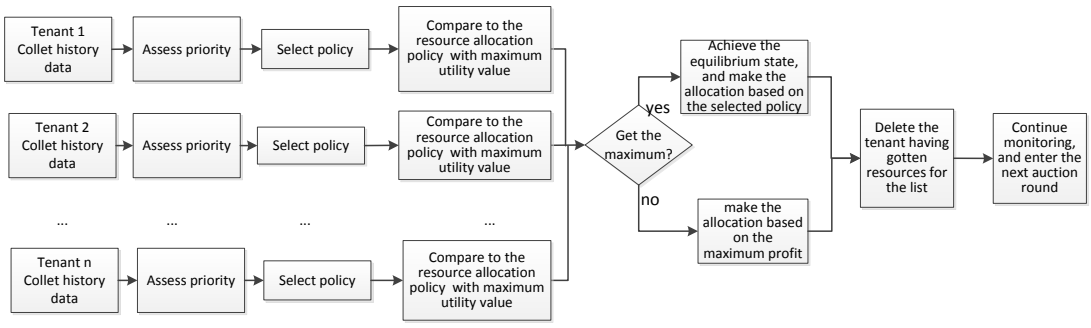


Figure 2. The Game Period

Based on the framework in Section 2, we take CPU as the main research object and

design the corresponding resource allocation strategy. Using the game model to solve the resource competition among multiple tenants, in essence, it is the problem that the tenants and the cloud service providers how to determine and select their policies. The tenant calculates the expected utility based on its SLA and resource emergency, and then sends the candidate policy and it's utility to the cloud service providers. The providers determine the policy that the overall utility is optimal and the game equilibrium is achieved. With the collected running data of tenant, such as CPU utilization, average response time of transaction and throughput of transaction, the future resource demand of the tenant is predicted and the priorities of the tenants are assessed. There exists the game period during the process of game, as shown in Figure 2. There are three major processes in the game period.

(1)**Process of resource prediction for multiple tenants.** By collecting and analyzing the historical data of the tenants' operation, we model and solve the transaction service time, and forecast the future resource demand of the tenant. It uses Kalman Filter to estimate the service time of each transaction based on monitoring the runtime data at fine-grained level, which has higher accuracy, compared with the way of directly using history data of resource consumption to forecast. For the data of each tenant, at first, the unobservable state x are modeled as a *N*-dimensional vector containing the service time of *N* transactions, that is $x_k = [S_1^k, S_2^k, \ldots S_N^k]$, where S_1^k represents the average service time of each transaction. According to the law of CPU utilization, the observed total CPU utilization u_k is defined as $u_k = \sum_{i=1}^{n} t_i \times s_i^k + v_k$, where t_i represents the throughput of each transaction at moment *k*, v_k represents the measurement error ($v_k = u_k - T_k * X_k$), and $T_k = [t_1, t_2, \ldots, t_n]$ indicates the throughput of each transaction. Then, the following iteration procedure based on Kalman filter is performed for each tenant.

(1) According to the state *x* at moment *k*-1, *x* at moment *k* is : $\hat{X}_k^- = \hat{X}_{k-1}$;

(2) According to the covariance at moment *k*-1, the covariance at moment *k* is estimated as:
$P_k^- = P_{k-1} + Q_k$;

(3) The Kalman gain K_k is calculated as: $K_k = P_k^- T_k^T (T_k P_k^- T_k^T + R_k)^{-1}$;

(4) The state *x* is updated by the observed variables U_k: $\hat{X}_k = \hat{X}_k^- + K_k(U_k - T_k \hat{X}_k^-)$;

(5) The covariance matrix is calculated as: $P_k = (1 - K_k T_k) P_k^-$.

The initial value is set for the iteration process as follows: $x_0 = [S_1^0, S_2^0, \ldots S_N^0] = [t_1 \times (1 - u_0), t_2 \times (1 - u_0), \ldots t_n \times (1 - u_0)]$, where u_0 is the CPU utilization of the system itself occupies when the tenants are not running. The above procedure will end after all the data in the monitoring window have been analyzed, and the predicted service time of transactions of each tenant T_s is obtained. Generally, the confidence interval of T_s is set to 95%, that is, the error of the estimated service time and the actual one should be less than 5%. The service time of each transaction is independent, so P_0 is set to a diagonal matrix: $P_0 = diag((S_1^0)^2, (S_2^0)^2, \ldots (S_n^0)^2)$. Q_k represents the change value of the covariance matrix at moment *k*, whose value is set to $Q_k = diag(\xi_1, \xi_2, \ldots, \xi_n)$, where $\xi_i = \max_{1 \le j \le k}((S_i^j - S_i^{j-1})^2)$ represents the square of the maximum change value of *x* in the iteration process, and R_k is the error of the measurement value of CPU utilization. Thus, the resource consumption of the tenant is predicted, and the number of expected resources for the tenants is also achieved.

(2) **Process of resource auction for multiple tenants.** The tenants put forward their own bidding policies according to their actual needs for resources. Therefore, it is

desirable to comprehensively consider the tenant's SLA and its resource usage so as to give priority to the tenants of high-SLA and high resource utilization, so as to reduce the occurrence of SLA violation. Firstly, it orders the tenants with the same level SLA in accordance with resource utilization, and then it prioritizes the tenants according to the different segmentation of resource utilization to come out a priority list. A list of tenants having bidding rights can be obtained based on the priority list and the available resource, and their biding policies are also gotten. As for the tenants of high resource utilization, it is required to meet their demand firstly in order to reduce SLA violation, at the same time the high-SLA tenants are ranked in the front so that the tenant's satisfaction can be further improved.

(3) **Process of resource allocation for multiple tenants**. The global game begins after the tenants give their bidding policies. A tenant have obtained its resource allocation policy P_{priority} according to the priority list in the above process, but these results may not reach the maximum utility of overall game, instead the waste of resources may occur. Therefore, the tenant's allocation policy under the maximum utility (a_i) is calculated by Formula (10) and (11), which is compared with P_{priority}, and the larger one will be the final policy. Unlike the general game strategy based on average allocation or reward/penalty mechanism, our strategy can not only meet the tenant's SLA but also be very close to the optimal utility of the global game by combining the consideration of SLA and resource utilization.

4. Experiments

This paper adopts transactional TPC-W benchmark [15] as tenant application for testing. In the experiments, the applications are set different SLA levels. The experiments are run on the virtual machines based on KVM. At first, the CPUs are assigned to the tenants according to their SLAs, and the CPU resources are adjusted online according to the game process. Multiple tenants run the TPC-W applications, and two kinds of tenants with different SLAs (high-level and normal-level) are initiated. The tenant has its own Web server and database server (Mysql as database) and all the tenants share the physical resources, as shown in Figure 3.

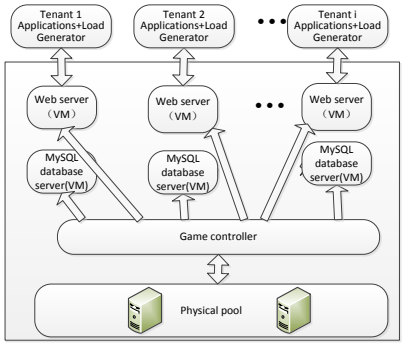


Figure 3. The Experiment Configuration

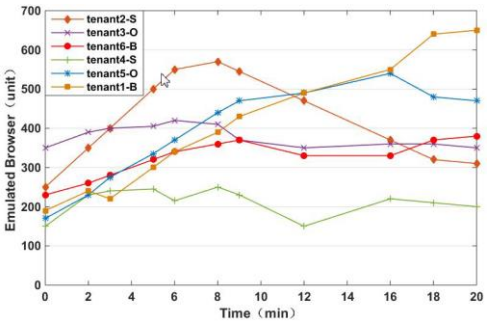


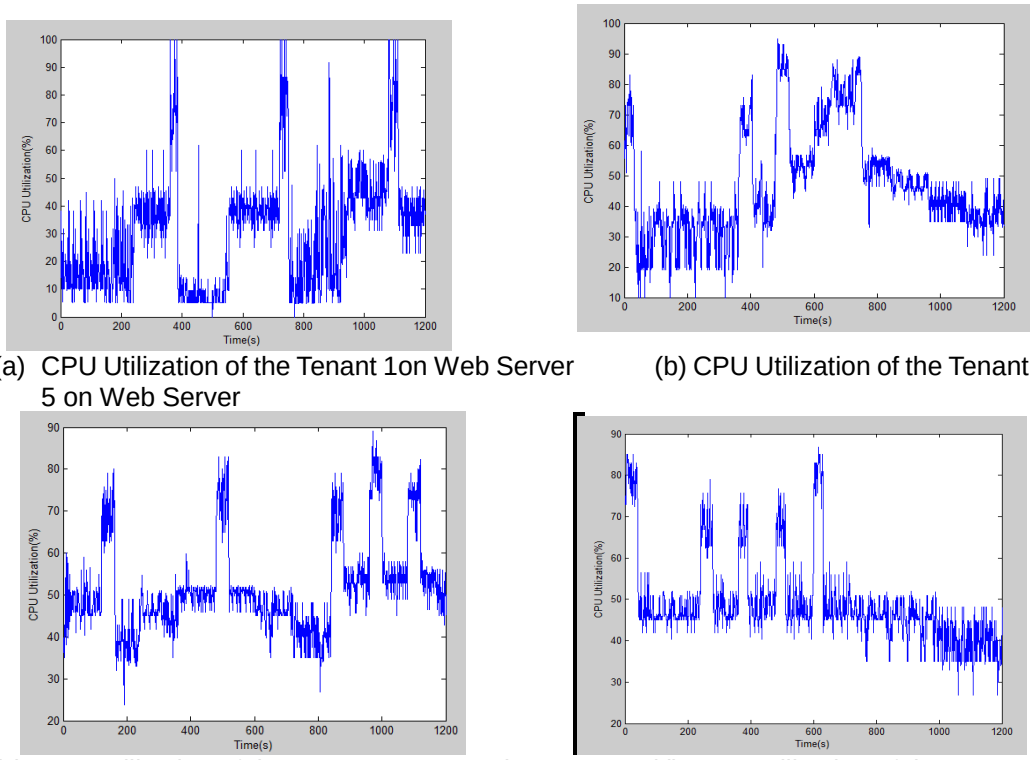
Figure 4. The Workload on the Tenants

We designed two groups of experiments. In the first group, the tenants are given a certain load and their CPU utilization, transaction's throughput, transaction's response time and the changes of resource allocation are monitored, in order that the resource bidding can be analyzed in case that there is resource competition. The second group of experiments is to compare our strategy to the game strategy based on reward/penalty in [16]. At the beginning of the experiments, the priority of each tenant is initially configured, i.e. tenant 1- 3 are high priority tenants, while tenant 4 - 6 are common priority tenants.

4.1 The Results of Experiment 1

In the first group of experiments, we put various load on the tenants. For instance, Figure 4 shows the workload on six tenants in the experiment. The ordinate represents the number of simulated TPC-W application browser (Emulated Browser, EB) run by tenant. Initially, the high priority tenants (tenant 1-3) are assigned 4 vCPUs (virtual CPU), and the common priority tenants (tenant 4-6) are assigned 2 vCPUs.

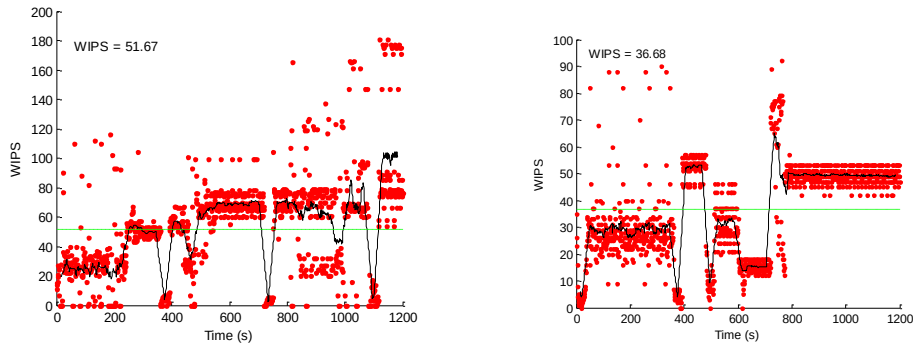
The CPU resource allocation by game under different load is analyzed as follows. As shown in Figure 4, the load of high-priority tenant 1 and common-priority tenant 5 show upward trends, and the needs for resource also have a tendency of rise. There may be multiple rounds of game for resources. Therefore tenant 1 and tenant 5 are selected as the observed objects. Then after 20 minutes' testing, the results are shown in Figure 5, Figure 6, and Figure 7.



(a) CPU Utilization of the Tenant 1on Web Server (b) CPU Utilization of the Tenant 5 on Web Server

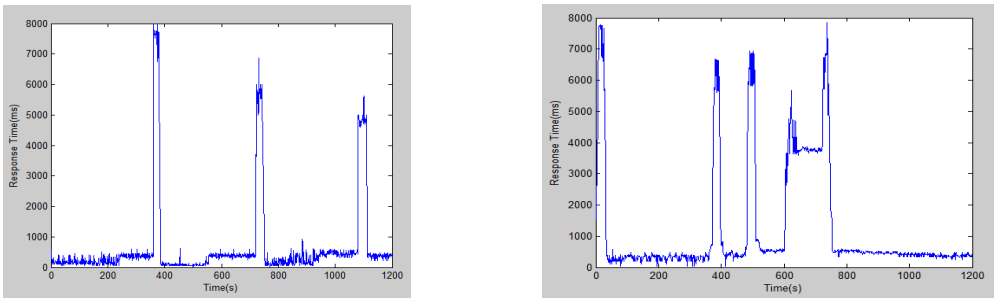
(c) CPU Utilization of the Tenant 1 on Mysql Server (d) CPU Utilization of the Tenant 5 on Mysql Server

Figure 5. The CPU Utilization



(a) Throughput of transaction of tenant 1
(b) (b) Throughput of transaction of tenant 5

Figure 6. The Transaction Throughput



(a) The Response time of Tenant 1
(b) The Response time of Tenant 5

Figure 7. The Response Time

As seen from Figure 5, the CPU utilization of Web server and database server both increase with the increasing workload of tenant 1 and tenant 5. Take the changes of CPU utilization of tenant 1 on Web server as the example, its CPU utilization increases to 100% when the workload increases to around 340 at the same time (see Figure 5(a)), and the throughput of tenant 1 is near to 0 (see Figure 6 (a)) and the response time increases to around 8 seconds sharply (see Figure 7(a)). All of them show that tenant 1 has strong demand for resources. As shown in Figure 8, the tenant gets 2 vCPUs by auction and the amount of vCPU becomes 6. And the system allocates 2 vCPUs to tenant 1, CPU utilization of tenant 1 decreases and the transaction throughput increases, at the same time, the response time decreases slowly. But not every change of workload leads to the request for resource game. Take another example, the change of CPU utilization of tenant 5 on Web server, initially it cannot deal with the situation when the working load reaches around 170 because of the small resources which have been allocated to tenant 5 (only 2vCPUs). Thus the game process starts at the beginning phase of the experiment. After the resource demand is satisfied, tenant 5 can deal with the subsequent workload well, and it will come into the next game round till the workload reaches about 370.

To sum up, according to the experiments and the figures above, as for tenant 1 and tenant 5, the times of game and the resource allocated after game are listed in Table 1.

Table 1. The Results of Game

Tenant id	Times of Game	Total resources in game	Average throughput(MPS)	Average response Time (s)
Tenant 1	8 times	19 vcpus	51.67	0.895
Tenant 5	9 times	17 vcpus	36.68	1.235

4.2 The Results of Experiment 2

In order to evaluate the efficiency of dynamic adjustment of tenant's CPU, we made a comparison between our strategy and the resource game strategy based on reward/penalty mechanism. The later strategy proposes a bidding model based on penalty mechanism for resource allocation, and deals with dynamic resource allocation by using Nash equilibrium of cooperative game. It analyzes the relationship between resource allocation and competition by perceiving the response time and the changes of CPU utilization. Under the same configuration as in experiment 1, the two kinds of strategy were carried out, and the

ISSN: 2005-4262 IJGDC

CPU utilization and allocated resource were compared. The results are shown in Figure 8.

By comparing Figure 5(a) and Figure 8, what can be seen is that the game allocation points of two strategies are anastomotic. After allocating resource by using the strategy based on reward/penalty mechanism, the resource utilization of tenant 1 is higher and it doesn't go down, which means that tenant 1 gets fewer resources and it may not meet its SLA. Thus, this strategy has the shortcoming in satisfying tenant's SLA. Figure 10 shows the tenant's resource allocation by the two strategies, which shows that our strategy is helpful for dynamically adjusting resources for the tenants, involving in the tenant's SLA and resource utilization.

The resource allocation strategy based on reward/penalty has short response time when processing game, and it has little influence on system performance. However, it gains better results merely according to Nash equilibrium without considering the SLA of tenants. As for the traditional fair resource competition game strategy, it can improve the tenant's service time by load balancing after creating new virtual machines and shutting down the virtual machines without affecting the running tenants' services. However, it cannot fit the personalized resource demand of tenants by just allocating resources averagely. Compared with the above two strategies, our strategy provides certain improvement on greatly satisfying multi-tenant's SLA and guarantee the QoS of resource provision.

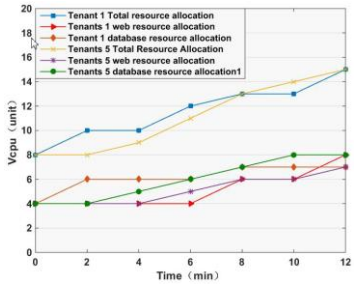


Figure 8. The Changes of Resources

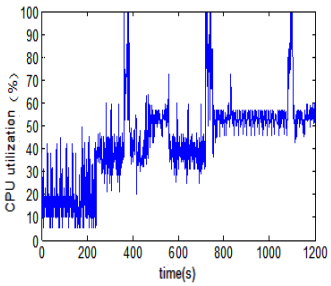


Figure 9. CPU Utilization of Tenant 1

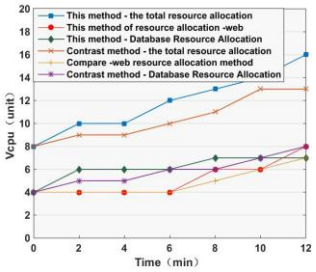


Figure 10. The Comparisons

5. Conclusion

The paper firstly analyses the feasibility of applying game theory to resource allocation for multi-tenant. Then a resource allocation strategy based on game model is presented, combining the consideration about tenants' SLA and their resource utilization, in order to deal with the shortcomings of the traditional allocation strategy that cannot satisfy the individual SLA demand and consider few about the urgent demand. The given experiments show the effectiveness of the presented strategy. The experimental results also indicates that it can meet the differentiated SLA demand of multi-tenant better, and helps to improve the system's resource utilization and decreases the operating costs of cloud service providers, which results in win-win. Some future work include the improvement of resource prediction by using more effective monitor and support for more kind of resource adjusting operations.

Acknowledgements

This work is supported by the Natural Science Foundation of China (No. 61063012, 61363003), the National Key Technology R&D Program of China (No. 2015BAH55F02).

References

[1] Buyya R, Garg S K, Calheiros R N. SLA-Oriented Resource Provisioning for Cloud Computing: Challenges, Architecture, and Solutions. In: Proceedings of 2011 International Conference on Cloud and Service Computing, 2011:1-10.
[2] Garg S K, Gopalaiyengar S K, Buyya R. SLA-based Resource Provisioning for Heterogeneous Workloads in a Virtualized Cloud Datacenter. In: Proceedings of the 11th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP 2011), 2011:371-384.
[3] Michael M, Ivona B, Vincent E. Towards Knowledge Management in Self-adaptable Clouds. In: Proceedings of IEEE 2010 Fourth International Workshop of Software Engineering for Adaptive Service-Oriented Systems, 2010:103-109.
[4] Rajkamal K G, Pushpendra K P. A Rule-based Approach for Effective Resource Provisioning. International Journal of Computer Science and Informatics, 2012, 1(4):2231-5292.
[5] Wei G, Vasilakos A V, Xiong N. Scheduling Parallel Cloud Computing Services: An Evolutional Game. In: Proceedings of 2009 1st International Conference on Information Science and Engineering, 2009: 376-379.
[6] Niyato D, Vasilakos A, Kun Z. Resource and Revenue Sharing with Coalition Formation of Cloud Providers: Game Theoretic Approach. In: Proceedings of IEEE/ACM International Symposium on Cluster Cloud and Grid Computing, 2011:215-224.
[7] Wei G, Vasilakos A, Zheng Y. A Game-theoretic Method of Fair Resource Allocation for Cloud Computing Services. The Journal of Supercomputing 2010, 54(2): 252-269.
[8] Giorgia L, Fabio P, Davide R. SLA-Driven Clustering of QoS-Aware Application Servers. IEEE Transactions on Software Engineering,2007,3(3):186-196.
[9] Meng X, Isci C, Kephart J, Zhang L, Bouillet E, Pendarakis D. Efficient Resource Provisioning in Compute Clouds via VM Multiplexing. In: Proceedings of the 7th international conference on Autonomic computing, 2010: 11-20.
[10] Hu D, Chen N, Dong S, Wan Y. A User Preference and Service Time Mix-aware Resource Provisioning Strategy for Multi-tier Cloud Services. AASRI Procedia, 2013,5: 235-242.
[11] Zheng M, Hu Z, Xiao P, Zheng M, Zhang K. A Global Benefit Maximization Task-bundle Allocation. In: Proceedings of the 8th IFIP international conference on Network and parallel computing, 2011: 71-85.
[12] Greg W, Gary B. An Introduction to the Kalman Filter. University of North Carolina, North Carolina, 2006.
[13] Wang W, Huang X, Qin X, Zhang W, Wei J, Zhong H. Application-Level CPU Consumption Estimation: Towards Performance Isolation of Multi-tenancy Web Applications. In: Proceedings of IEEE 5th international conference on Cloud computing. 2012: 439-446.
[14] Gibbons R. A Primer in Game Theory. Harvester Wheatsheaf, 1992.
[15] http://www.tpc.org/tpcw/,2015.9.10
[16] Jiang C, Duan L, Liu C, Wan J, Zhou L. VRAA: Virtualized Resource Auction and Allocation Based on Incentive and Penalty. Cluster computing, 2013, 16(4), 639-650.

Authors

Chen Ningjiang, born in 1975. Professor and PhD. His main research interests include distributed computing and software engineering.

Tan Ying, born in 1990. Received her bachelor degree in Guangxi University. Her main research interest is distributed computing.

Li Xiang, born in 1989. Received her bachelor degree in Guangxi University. Her main research interest is cloud computing

Liang Xiaoyu, born in 1991. Received her bachelor degree in Guangxi University. Her main research interest is software engineering.

Huang Ruwei, born in 1978. Associate professor and PhD. Her main research interests include cloud computing and information security.