DOI: http://dx.doi.org/10.26483/ijarcs.v9i1.5389

Volume 9, No. 1, January-February 2018

ISSN No. 0976-5697

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

IMPLEMENTING TASK AND RESOURCE ALLOCATION ALGORITHM BASED ON NON-COOPERATIVE GAME THEORY IN CLOUD COMPUTING

G.Prasadu
Assistant Professor, Dept of IT, SNIST,
Hyderabad,India

Dr.P.Ila Chandana Kumari
Professor, Dept of CSE,HITAM,
Hyderabad,India

Dr.P.Gayathri
Associate Professor, Dept of IT,GRIET,
Hyderabad,India

Abstract: Now a day's cloud computing is becoming most popular computing model in the information industry, and cloud is a large and complex system with a more number of servers and users, so it is necessary to manage all the tasks and users depending on the traffic of the network. To manage tasks and users it has to schedule tasks frequently among the servers and manage its computing resource flexibly to meet the demand of the users. With the growing service demand and higher QoS requirement of the users, the performance of the system is facing a big challenge, and with the expanding scale of cloud computing, its energy waste problem is becoming more and more serious due to the invalid resource organization and failed task scheduling. To improve the energy efficiency of heterogeneous servers in the cloud computing system, this paper puts forward a non-cooperative game based task scheduling and computing resource allocation algorithm NG_TSRA. Firstly, we use non-cooperative game to model the task scheduling and computing resource allocation process of the servers in the cloud computing system, and the server's utility function is modeled as unit power efficiency, then we prove the existence of Nash Equilibrium point of the game, and finally use a Lagrange multiplier-based distributed iteration algorithm to solve the game. The experimental results show that the proposed algorithm can improve the average power efficiency of the cloud computing system.

Keywords: Non-Cooperative game, task scheduling, resource scheduling, cloud computing, efficiency, heterogeneous

1. INTRODUCTION

The cloud is widely used and becomes the academic research hotspot in recent years, Cloud Service Providers (CSPs) have provided solutions for various internet services such as electronic commerce, 020, medical care, online game, finance, content delivery and so on, the users also have higher QoS demand than before which requires a higher bandwidth and lower latency service. In addition, the energy consumption is becoming a bottleneck which limits the scale and service capacity of the cloud. As cloud computing carries more and more network services, its system load is very large, the task of the system is divided into many small sub-task sequences, and then processed by various distributed cloud servers collaboratively. In addition, cloud computing provides a flexible resource allocation scheme, it can allocate computing resource according to the amount of tasks, then the servers will have higher resource utilization and lower energy consumption. As the current scale of cloud computing system is expanding, there are all kinds of cloud servers with different performance parameters and the cloud service resource is heterogeneous. Therefore, how to accomplish task scheduling and computing resource allocation efficiently in the complex cloud system becomes one important topic of cloud computing.

A lot of researches have been done on task scheduling and resource allocation in cloud computing system. In [I], a resource scheduling framework for mobile cloud computing system based on nested two-stage game is proposed, which effectively controls the access load of the system and reduces the user's response delay. Aiming at reducing the uncertainty of task execution time in IaaS cloud service model, [2] proposes a dynamic resource allocation a nd task scheduling algorithm, which can reduce the extra cost caused by the uncertainty of task execution time. [3] proposes a self-adaptive cloud computing resource management mechanism, which can maintain the system performance while improving the utilization of the hardware resources of the cloud computing system. From the point of view of
market relations. [4] proposes a bidirectional auction-based cloud resource allocation algorithm, which effectively protects the interests of buyers and sellers in the bidding process of computing resources. [5] proposes a cloud computing task scheduling method based on dynamic adaptive ant colony algorithm, which greatly improves the task scheduling time efficiency. Taking into account task dependency, the work [6] puts forward a Dependency-aware and Resource-efficient Scheduling scheme (DRS) to achieve a low response time and high resource utilization task scheduling. In addition, there are some task scheduling algorithms designed to shorten the task completion time of cloud computing system [7-9].

Although there are lots of researches on resource allocation and task scheduling of cloud computing system, most of them focus on how to improve user's service quality, rarely considering system energy consumption. With the current scale of cloud computing system continues to expand, its energy consumption problems are becoming more and more prominent, and get extensive attention by the industry and government agencies [10]. What's more, when

taking the energy consumption into consideration, how to reduce the power consumption while guaranteeing the users' QoS in the heterogeneous cloud environment becomes a new difficulty. In this paper, non-cooperative game is involved to solve the task scheduling and resource allocation problem in the cloud computing system with heterogeneous servers. Aiming at optimizing the unit power efficiency of the servers, we let the unit power benefit be the utility of the servers, the server will regulate its own amount of tasks and determine the optimal computing resources proportion to maximize
Utility and finally achieves a strategy balance among the whole system.

The rest of this paper is organized as follows. Section II Discusses about the related work done on the area of scheduling in cloud computing. Section III introduces the model of cloud computing system and discusses its energy efficiency problems, then models the system as a non-cooperative game and proves the existence of the Nash equilibrium point. A Lagrange multiplier-based distributed iteration algorithm is proposed to solve the game model in Section IV. Experimental results are presented in Section V, and we conclude this paper in Section VI.

2. RELATED WORK

Wang Y et al[1] elaborated the game theory for optimizing mobile cloud computing resource in A nested two stage game-based optimization framework in mobile cloud computing system. Liu S [2] discussed about dynamic resource allocations strategies in uncertain runtime environments in IaaS services in cloud computing.Huang C et al[3] proposed an excellent algorithm for an adaptive resource management scheme in cloud computing. Ma T and Xie R [4] proposed a scheme for allocating cloud computing resources based on double auction considering interests of both buyers and sellers under cloud computing environment. Wang.F et al [5] designed a task scheduling based on dynamically adaptive ant colony algorithm in cloud computing environment. Liu.J et al[6] proposed a co operative opportunistic resource provisioning algorithm for short term resources in cloud computing systems.
Liu Y et al [8] designed an algorithm based on genetic and ant colony for scheduling of tasks in cloud computing environment. Shark MA[10] made an excellent survey on design challenges in resource allocation in network based cloud computing. Susanto H et al[11] developed a model in cloud computing for Congestion Control with QoS and Delays Utility Function.Mukundha.ch et al[15] discussed on scheduling algorithms using software defined networks in cloud computing.

3. PROBLEM DESCRIPTION AND SYSTEM MODELING

A. Problem Description

Consider a cloud computing system with heterogeneous servers as shown in Figure 1, which consists of a service request pool, a central scheduling node, and servers of N types, and each type with M_N servers. Service request pool is used for storing the service request from all users from the same application, the total service request is λ. Central scheduling node is responsible for assigning the user service requests to the various cloud computing servers and collecting the resource scheduling information. We define the j^{th} server of the i^{th} type as server S_{ij} . The performance parameters of the servers are different due to different types, task proportion of server S_{ij} is denoted by $0< P_{ij} < 1$, then the amount of task it needs to deal with is λP_{ij} . At the same time, the proportion of resource it allocated to process the user's requests is $0< \Phi ij < 1$, if server S_{ij} use all his resource to process its tasks, set $\Phi ij = 1$, meanwhi le its power consumption will be the highest. When all the servers select the optimal task proportion and match the optimal computing resource proportion to process it, the power efficiency can be maximized. Not only can we reduce the power consumption to the maximum extent but also process the user requests located on the server at a faster speed.

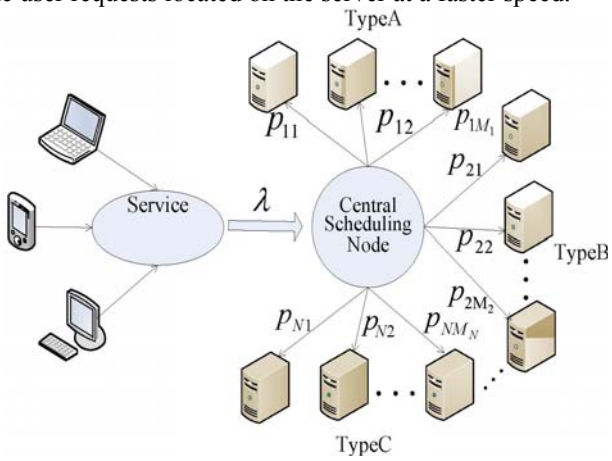

Figure 1: A cloud model with heterogeneous servers

Assuming that the overall task of the system far exceeds the processing capacity of a single server, multiple types of cloud servers need to cooperate to deal with it. While the available computing resource of each server is limited, let μ_{ij} denote the maximum processing speed of sever S_{ij} when all its resource is used to process user requests. P_{ij} is the proportion of task that is targeted to server S_{ij} . Φ_{ij} is the portion of its total resource that server Sij allocates to process its tasks, so its current processing speed is $\Phi_{ij} .\mu_{ij}$. Then the service request response delay of S_{ij} can be expressed by equation (1) [11]:

$$R_{ij} = \frac{1}{\phi_{ij} \cdot \mu_{ij} - p_{ij} \cdot \lambda} \qquad (1)$$

Assuming that the task they choose does not exceed the processing capacity of their own, which means $\Phi_{ij}.\mu_{ij} > P_{ij}\ \lambda$. Then the utility of server S_{ij} related to user delay is expressed as [11]:

$$v(R_{ij}) = \lg(\frac{1}{R_{ij}}) = \lg(\phi_{ij} \cdot \mu_{ij} - p_{ij} \cdot \lambda) \qquad (2)$$

The utility function of the cloud computing server S_{ij} related to user's Quality of Service (QoS) is modeled as:

$$\hat{U}_{ij} = \lambda p_{ij} v(R_{ij}) = \lambda p_{ij} \lg(\phi_{ij} \cdot \mu_{ij} - p_{ij} \cdot \lambda) \qquad (3)$$

The power consumption of server S_{ij} is composed of two parts, including idle power consumption denoted by $\square_{idle,ij}$

and peak power consumption denoted by $P_{peak,ij}$ the server's power consumption is related to the proportion of resource it allocates. When no task is to be processed, Φ_{ij} will be 0, and there is only idle power consumption. Then use standard linear function to model the server's power consumption [12]:

$$P_{Serv,ij}(\phi_{ij}) = \varepsilon_{idle,ij} + \phi_{ij} \cdot (P_{peak,ij} - \varepsilon_{idle,ij}) \quad (4)$$

In order to improve the energy efficiency of the whole cloud computing system, the server's power efficiency is modeled as unit power utility:

$$U_{ij} = \frac{\hat{U}_{ij}}{P_{Serv,ij}(\phi_{ij})} = \frac{\lambda p_{ij} \lg(\phi_{ij} \cdot \mu_{ij} - p_{ij} \cdot \lambda)}{\varepsilon_{idle,ij} + \phi_{ij} \cdot (P_{peak,ij} - \varepsilon_{idle,ij})} \quad (5)$$

B. Non-Cooperative Game Modeling

The cloud computing task scheduling and resource allocation model is denoted by G={X, $\{P_{ij},\Phi_{ij}\}$,{ U_{ij}},i □ N, j □ M_N} and can be treated as a non-cooperative game. The players of this game are all the heterogeneous servers denoted by □ , the strategy of player S_{ij} is (p_{ij},Φ_{ij}), and its utility is unit power efficiency U_{ij}.In this game model, the server S_{ij} needs to decide two strategies: one is the task proportion P_{ij} subscribed from the scheduler, the other is the computing resources proportion Φ_{ij} it allocated to process this task. Assuming that the total request λ remains constant at present, in order to maintain their own profit, each server will regulate its own proportion of task and the proportion of computing resource allocated according to its own processing capability, power consumption and other servers' strategies. If total request λ is always smaller than the maximum carrying capacity of the whole cloud computing system, each server will regulate its strategy in order to maximize its own utility. The strategies of the servers will influence each other because of the constraint of total task λ, this can be treated as a non-cooperative game process. For each sever, putting more computing resource means more power consumption, but it can promote user's QoS, thus it has an optimal value. The utility of server S_{ij} is directly determined by Φ_{ij} and P_{ij} in order to achieve higher unit power efficiency, the values of these two parameters must match. Assuming that once a server updates its strategy, it will upload information to the central task scheduler, and then the game information will be sent to the other servers by the central scheduling node to share the information globally and ensure the equality of game information among the game players. The concrete resource scheduling process can be shown in Figure 2:

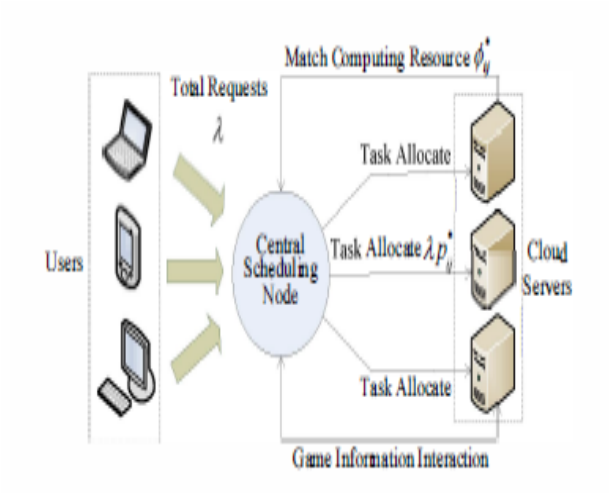

Figure 2: Non-Cooperative scheduling System in Cloud Computing

In this game model, the final goal of each server is to find the best task proportion and the matching resource allocation proportion to maximize the power efficiency of it. This problem can be expressed as equation (6):

$$(\phi_{ij}^*, p_{ij}^*) = \arg\max U_{ij} \quad (6)$$
$$\text{s.t.} \sum_{i=1}^{N} \sum_{j=1}^{M_N} p_{ij} = 1$$
$$0 \le \phi_{ij} \le 1$$
$$0 < p_{ij} < 1$$

Each server's strategy will change with the other server's strategy changes under the total task constraint which is denoted by the first constraint in formula (6). If all the servers respond best to the strategies of each other, the game model will reach an equilibrium state, which is the Nash equilibrium point. At this point, all the servers of the system gain their best utilities, and no one can get more revenue by changing their strategy privately.

C. The Existence a/Nash Equilibrium

Then prove the existence of the Nash equilibrium point of this game model.

Theorem I: When the resource proportion of each server Φ={ $\Phi_{11},\Phi_{12},\ldots\Phi_{nm}$} is determined, servers try to optimize their own unit power efficiency, this process can be treated as a non-cooperative game, and this game model has Nash equilibrium denoted by $P^* = \{P^*_{11},P^*_{12},\ldots..P^*_{nm}\}$.

Proof : Respectively, the first derivative and second derivative of utility function in (5) with respect to P_{ij} is as in (7) and (8):

$$\frac{\partial U_{ij}}{\partial p_{ij}} = \frac{1}{\ln 10 \cdot P_{Serv,ij}(\phi_{ij})} [\lambda \ln(\phi_{ij}\mu_{ij} - p_{ij}\lambda) - \frac{\lambda^2 p_{ij}}{\phi_{ij}\mu_{ij} - p_{ij}\lambda}] \quad (7)$$

$$\frac{\partial^2 U_{ij}}{\partial p_{ij}^2} = -\frac{1}{\ln 10 \cdot P_{Serv,ij}(\phi_{ij})} [\frac{2\lambda^2}{\phi_{ij}\mu_{ij} - p_{ij}\lambda} + \frac{\lambda^3 p_{ij}}{(\phi_{ij}\mu_{ij} - p_{ij}\lambda)^2}] < 0 \quad (8)$$

It's obvious that Pij > 0 ,λ> 0 , $\Phi_{ij}\mu_{ij}$ - $P_{ij}\lambda$ > 0 , thus the utility function of the server satisfies the property of the concave function with respect to P_{ij} , and there are

$\left.\frac{\partial U_{ij}}{\partial p_{ij}}\right|_{p_{ij}=0} > 0$ and μij=λ we can get

$$\omega = \frac{\phi_{ij}\mu_{ij}}{\lambda} < 1 \text{ and } \lim_{p_{ij}\to\omega^-} \frac{\partial U_{ij}}{\partial p_{ij}} < 0$$

So there exists and extreme point p^*_{ij} between 0 and □ which maximizes the utility function.

Theorem II: When the server's workload P_{ij} is determined, each server can always find a proportion of computing resources Φ^*ij that match with it, and it satisfies $0 \le \Phi_{ij} \le 1$.

Proof : The first derivative of utility function in (5) with respect to Φ_{ij} is as in (9):

$$\frac{\partial U_{ij}}{\partial \phi_{ij}} = \frac{\lambda p_{ij}}{\ln 10 \cdot P^2_{Serv,ij}(\phi_{ij})}\left[\frac{\mu_{ij} P_{Serv,ij}(\phi_{ij})}{\phi_{ij}\mu_{ij} - p_{ij}\lambda} - \ln(\phi_{ij}\mu_{ij} - p_{ij}\cdot\lambda)(P_{peak,ij} - \varepsilon_{idle,ij})\right] = 0 \quad (9)$$

Let $\delta_{ij} = \frac{\mu_{ij} P_{Serv,ij}(\phi_{ij})}{\phi_{ij}\mu_{ij} - p_{ij}\lambda} - \ln(\phi_{ij}\mu_{ij} - p_{ij}\cdot\lambda)(P_{peak,ij} - \varepsilon_{idle,ij})$, and then can get:

$$\frac{\partial \delta_{ij}}{\partial \phi_{ij}} = -\frac{\mu^2_{ij} P_{Serv,ij}(\phi_{ij})}{(\phi_{ij}\mu_{ij} - p_{ij}\lambda)^2} < 0 \quad (10)$$

It can be seen that the utility function of the server to meet the characteristic of concave function with respect to Φ_{ij}, so in the interval [0,1] we can always find one Φ^*_{ij} to maximize its utility, and the value of Φij will match with the proportion of task P_{ij} it chooses. Then we can see this game model is convex [13]. And the strategies of the servers will converge to the Nash equilibrium point.

4. ALGORITHM DESCRIPTION

Because the model has two optimization goals, and it has equality constraint for PII ' thus this paper proposes a Lagrange multiplier-based distribution iteration algorithm to solve the Nash equilibrium of the game model. Firstly, the game problem is transformed into a conditional extreme problem. The Lagrange multiplier is used, and the constructed function is as follows:

$$L_{ij}(p_{ij},\phi_{ij}) = \frac{\lambda p_{ij} \lg(\phi_{ij}\mu_{ij} - \lambda p_{ij})}{P_{Serv,ij}(\phi_{ij})} - \tau\left(\sum_{i=1}^{N}\sum_{j=1}^{M_N} p_{ij} - 1\right) \quad (11)$$

τ > 0 is a Lagrange multiplier, then we use KKT condition and the partial derivative of the Lagrange function Lij (P_{ij},Φ_{ij}) in (11) on P_{ij},Φ_{ij} and τ is as in (12), (13) and (14):

$$\frac{\partial L_{ij}}{\partial \phi_{ij}} = \frac{\lambda p_{ij}}{\ln 10 \cdot P^2_{Serv,ij}(\phi_{ij})}\left[\frac{\mu_{ij} P_{Serv,ij}(\phi_{ij})}{\phi_{ij}\mu_{ij} - p_{ij}\lambda} - \ln(\phi_{ij}\mu_{ij} - p_{ij}\cdot\lambda)(P_{peak,ij} - \varepsilon_{idle,ij})\right] = 0 \quad (12)$$

$$\frac{\partial L_{ij}}{\partial p_{ij}} = \frac{1}{\ln 10 \cdot P_{Serv,ij}(\phi_{ij})}\left[\lambda \ln(\phi_{ij}\cdot\mu_{ij} - p_{ij}\cdot\lambda) - \frac{\lambda^2 p_{ij}}{\phi_{ij}\mu_{ij} - p_{ij}\lambda}\right] - \tau = 0 \quad (13)$$

$$\frac{\partial L_{ij}}{\partial \tau} = \sum_{i=1}^{N}\sum_{j=1}^{M_N} p_{ij} - 1 = 0 \quad (14)$$

So we can solve p_{ij} from (13) as in (15):

$$p_{ij} = \frac{1}{\ln 10 \cdot P_{Serv,ij}(\phi_{ij})\tau}\left[\lambda p_{ij} \ln(\phi_{ij}\mu_{ij} - p_{ij}\lambda) - \frac{\lambda^2 p^2_{ij}}{\phi_{ij}\mu_{ij} - p_{ij}\lambda}\right] \quad (15)$$

Then get τ from (14) and (15) as in (16):

$$\tau = \sum_{i=1}^{N}\sum_{j=1}^{M_N} \frac{1}{\ln 10 \cdot P_{Serv,ij}(\phi_{ij})}\left[\lambda p_{ij} \ln(\phi_{ij}\mu_{ij} - p_{ij}\lambda) - \frac{\lambda^2 p^2_{ij}}{\phi_{ij}\mu_{ij} - p_{ij}\lambda}\right] \quad (16)$$

Firstly, we define a sub-iteration process :

First step: When the value of τ is set, and the values of $\Phi_{ij}(k)$ and $P_{ij}(k)$ of each server are initialized, substitute $\Phi_{ij}(k)$ into equation (13) to solve $P_{ij}(k+l)$, and substitute $p_{ij}(k+1)$ into equation (12) to solve Φij(k+l), then use these two values to update the server's task scheduling proportion and resource investment proportion, if $\Phi_{ij} > 1$, set $\Phi_{ij} = 1$.

Second step: Continue the calculation process above similarly to get $P_{ij}(k+2)$ and $\Phi_{ij}(k+2)$.

Third step: Do the iteration above until p_{ij} (k+2) -P_{ij} (k+ 1)< □ , stop the iteration and treat the eventually iteration results $p_{ij}(k+2)$ and $\Phi_{ij}(k+2)$ as the optimal task scheduling proportion and computing resource proportion under the value of τ at present, denoted by $P^*_{ij}$and Φ^*_{ij}respectively, then the sub-iteration is over.

The whole iteration process is as follows:

First step: Set the value of τ at time t, denoted by τt) , then initialize p_{ij}(t) and $\Phi_{ij}(t)$, and execute the sub-iteration process, let its final results p_{ij} and Φ_{ij} be the optimal task and optimal resource proportion at time t + 1, denoted by p^*_{ij}(t+ 1) and Φ^*_{ij} (t+1) respectively, then substitute them into equation (17) to solve τ(t + 1) .

Second step: Set τ = τ(t + 1) at iteration time t+ 1. Let p^*_{ij}(t+1) and Φ^*_{ij}(t+l) be the initial value of sub-iteration process, execute the sub-iteration process again to update the optimal tasks proportion p^*_{ij}(t+ 2) and computing resource proportion Φ^*_{ij}(t + 2), and use (17) to update τ(t + 2).

Third step: Repeat the process above until | τ(t+2) –τ(t+ 1)|< □, stop the whole iteration and let the eventual iteration results p_{ij}(t + 2) and Φ_{ij}(t + 2) be the final optimal task scheduling proportion and computing resource proportion, and t+2 be the final iteration time.

$$\tau(t+1) = \sum_{i=1}^{N}\sum_{j=1}^{M_N} \frac{1}{\ln 10 \cdot [\varepsilon_{idle,ij} + \phi^*_{ij}(t)\cdot(P_{peak,ij} - \varepsilon_{idle,ij})]}\left[\lambda p^*_{ij}(t)\cdot \ln(\phi^*_{ij}(t)\cdot\mu_{ij} - p^*_{ij}(t)\cdot\lambda) - \frac{\lambda^2 p^{*2}_{ij}(t)}{\phi^*_{ij}(t)\mu_{ij} - p^*_{ij}(t)\lambda}\right] \quad (17)$$

5. EXPERIMENT AND THE RESULTS

A. Experimental Environment

This experiment assumes that there are three types of servers, each type has 10 servers. The performance parameters (power consumption, maximum processing speed) of the three types of servers are different. The simulation parameters are as follows:

Table I: Server Performance Parameters

Performance parameters	Type A	Type B	Type C
Maximum processing speed μ_{ij} (MB/S)	15	22	35
Peak power $P_{serv,ij}^{phy,max}$ (W)	270	330	350
Idle power $\varepsilon_{serv,ij}$ (W)	25	30	40

Then, set λ=250MB / s, assuming that three types of servers initial computing resource proportion are all 1,
which means initialize Φ_{ij} =1 . In the first time, the initial workload of three types of servers are all 0.01, which means p_{ij} =0.01, and initialize τ=1, the experimental results are shown in the results section.

B. Experimental Results

Figure 3 shows the iteration process of the task proportion of the three servers from Type A, Type B, and Type C. As the total amount of task is certain, there is a non-cooperative game relationship between the servers, their strategies affect each other, and finally the task proportion of each server tends to converge. It can be seen from the figure that the task proportion of the sever from type A, type B, type C converges to 2.107%, 2.765% and 5.128% respectively, and the total task is all allocated among 30 servers. The task proportion of each server reaches convergence after 26 iterations, and no server can break the equilibrium by changing its own strategy privately, which achieves the optimal allocation of tasks among all the heterogeneous servers.

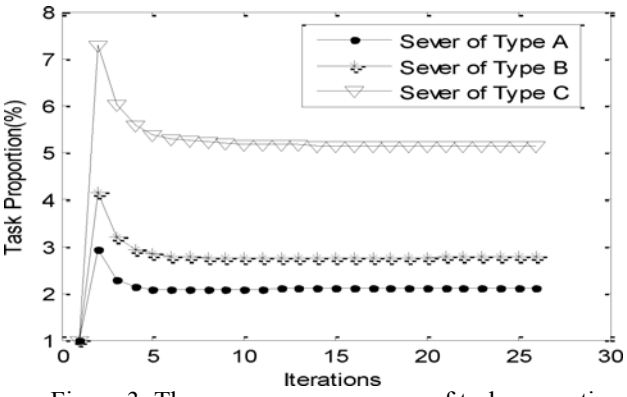

Figure 3: The convergence process of task proportion.

Figure 4 shows the iteration process of the resource proportion of the three servers, we set the initial resource allocation proportion of the three servers to 100%, but the three types of servers are all not fully loaded, and at the same time, due to the change of the workload at different time and the consideration of power efficiency, the three servers regulates their own resource allocation proportion and closes a part of their own computing resource. In the whole process, the servers compete the limited amount of tasks and decide the best task proportion according to p_{ij} and then its choice of task will affect their choice of resource proportion in turn.

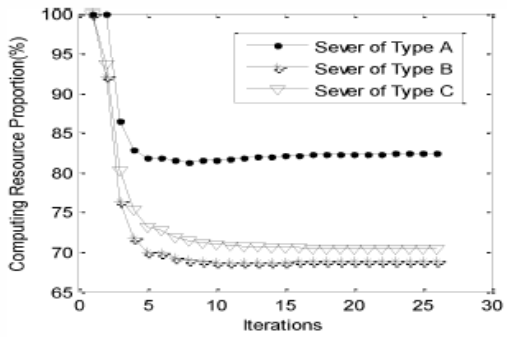

Figure 4: The Convergence Processes of Resource Proportion

In addition, its decisions are also affected by other servers strategies indirectly. During the whole iteration process, each server tries to optimize its own unit power efficiency, and make the resource proportion match with its task proportion. Similarly, the resource allocation proportion of the three servers converge after 26 iterations, and the proportion of resource proportion varies due to the difference in performance parameters between heterogeneous servers. Figure 5 shows the variation of the unit power efficiency of the three servers. It can be seen from the figure 5, in the initial state, the three servers of the unit power efficiency are very low, this is mainly caused by the mismatch of computing resources and the task amount, then the three servers change their strategies. When the total task is constant, the unit power efficiency of a single server will not increase or decrease unlimitedly due to the selfishness of each server in the non-cooperative game, the strategy constraint among the servers and the limitation of its own performance parameters. As can be seen from the figure, after 26 iterations, the unit power efficiency of the three servers achieves a constant value, it is determined by p^*_{ij} and Φ^*_{ij} together, then the strategies of the heterogeneous servers in the cloud computing environment reaches the Nash equilibrium.

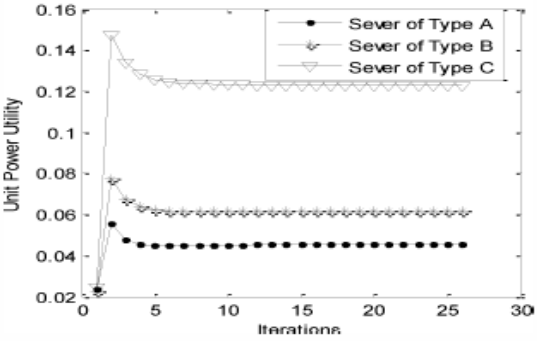

Figure 5 : The Convergence processes of Unit Power Efficiency.

The average power efficiency is the ratio of the total system revenue

$$\sum_{i=1}^{N}\sum_{j=1}^{M_N}\hat{U}_{ij}$$

to the total power,

$$\sum_{i=1}^{N}\sum_{j=1}^{M_N} p_{ij}$$

which reflects the energy efficiency level of the whole system. In order to prove the effectiveness of this algorithm, the average power efficiency of this algorithm is compared with other algorithms. Among them, the algorithm I do the task allocation according to the maximum processing capacity of the server, that is

$$p_{ij} = \frac{\mu_{ij}}{\sum_{i=1}^{N}\sum_{j=1}^{M_N}\mu_{ij}}$$

the task proportion is determined according to the servers maximum processing capacity, and then calculate the optimal resource proportion they should allocate of each server. Algorithm 2 is an algorithm proposed in [14], that is, according to the user's resource allocation proportion to calculate the proportion of the task, the algorithm directly build the relationship between p_{ij} and Φ_{ij} The NG_TSRA algorithm proposed in this paper try to determine the appropriate amount of task for every server under total task constraint and match p_{ij} with Φ_{ij} , so that the server power efficiency is optimized.

It can be seen from Figure 6, when λ gets different values, the average power efficiency of the algorithm proposed is higher than the other two algorithms, and when λ is greater than 480MB / S, due to the constraint of user's QoS, the unit power efficiency of algorithm 1 and algorithm 2 is reduced, but the algorithm proposed can guarantee the unit power utility of the servers. This is because it controls the total amount of task accessed to the cloud system, that is, when the task is overload, the value of τ in the Lagrange multiplier algorithm in this paper tends to 0,then the total task amount constraint becomes extremely weak, and the system implements a mode which make best effort to process as much task as possible when guarantee the server's unit power efficiency, the servers maintain the best efficiency by remaining a part of task to process in the next turn or dispatch it to other more servers, so the algorithm proposed can ensure the user's QoS. The experimental results prove the effectiveness of the proposed algorithm for task scheduling and resource allocation in the cloud computing environments.

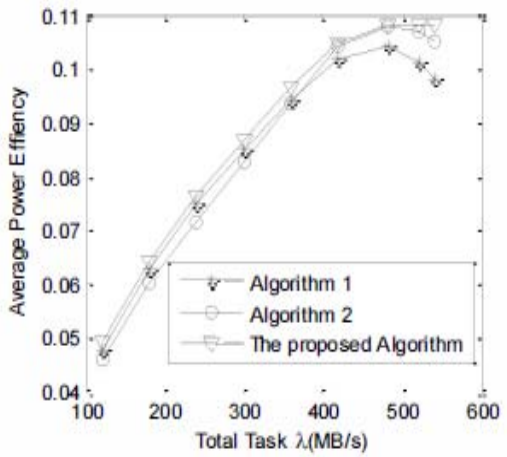

Figure 6: The Comparison of system power efficiency

6. CONCLUSIONS

Cloud computing has become the most popular computing model at present, and various network services are moving to the cloud. In this paper, a non-cooperative game-based cloud computing task scheduling and computing resource allocation algorithm NG_TSRA is proposed. This algorithm models the relationship of heterogeneous servers in the cloud computing system as a non-cooperative game. In the case that the total system request amount is a constraint, each server chooses the appropriate task quantity and matches the optimal computing resource proportion to deal with the user request according to its performance parameter characteristics and other servers strategies, aiming at minimizing their own energy consumption when the task of the system is efficiently processed, it can achieve higher power efficiency. In order to guarantee the existence of the optimal task proportion and the optimal resource proportion, this paper proves the existence of the Nash equilibrium of this game, and proposes a Lagrange multiplier-based distribution iteration algorithm to solve the Nash equilibrium, the effectiveness of the algorithm is proved by comparison with the existing algorithms. In order to obtain higher energy efficiency, the next step of this paper is to study a task scheduling and resource allocation algorithm based on NG_TSRA which can dynamically regulates the number of servers that participate in task processing based on sleep mechanism.

REFERENCES

[1] Wang Y, Lin X, Pedram M. A nested two stage game-based optimization framework in mobile cloud computing system[ClIIService Oriented System Engineering (SOSE), 2013 IEEE 7th International Symposium on. IEEE, 2013: 494-502.

[2] Liu S, Ren K, Deng K, et al. A dynamic resource allocation and task scheduling strategy with uncertain task runtime on laaS clouds[CllIInformation Science and Technology (ICIST), 2016 Sixth International Conference on. IEEE, 2016: 174-180.

[3] Huang C J, Guan C T, Chen H M, et al. An adaptive resource management scheme in cloud computing[J]. Engineering Applications of Artificial Intelligence, 2013, 26(1): 382-389.

[4] Ma T, Xie R, Liao F. Resource allocation algorithm based on double auction considering interests of both buyers and sellers under cloud computing environment [J]. Application Research of Computers, 2016, 33(3): 734-740.

[5] Wang F, Li M, Duan W. Cloud computing task scheduling based on dynamically adaptive ant colony algorithm [1]. Journal of Computer Applications, 2013, 33(11): 3160-3162.

[6] Liu J, Shen H, Chen L. CORP: Cooperative opportunistic resource provisioning for short-lived jobs in cloud systems[ClIICluster Computing (CLUSTER), 2016 IEEE International Conference on. IEEE, 2016: 90-99.

[7] Xue J, Li L, Zhao S, et al. A Study of Task Scheduling Based on Differential Evolution Algorithm in Cloud Computing[ClIIComputational Intelligence and Communication Networks (CICN), 2014 International Conference on. IEEE, 2014: 637-640.

[8] Liu C Y, Zou C M, Wu P. A task scheduling algorithm based on genetic algorithm and ant colony optimization in cloud computing[ClIIDistributed Computing and Applications to Business, Engineering and Science (DCABES), 2014 13th International Symposium on. IEEE, 2014: 68-72.

[9] Deng J, Zhao Y, Yuan H, et al. Multi-QoS objective constrained task scheduling strategy of cloud computing [J]. Application Research of Computers, 2016, 33(8): 2479-2482.

[10] Sharkh M A, Jammal M, Shami A, et al. Resource allocation in a network -based cloud computing environment: design challenges [1]. IEEE Communications Magazine, 2013, 51(11): 46-52.

[11] Susanto H, Kim B G. Congestion Control with QoS and Delays Utility Function[ClllICCCN. 2013: 1-5.

[12] Barroso L A, Holzle U. The Case for Energy-Proportional Computing[J]. Computer, 2007, 40(12):33-37.

[13] Liu J, Shen H. A low-cost multi-failure resilient replication scheme for high data availability in cloud storage[ClIIProc. of Hi pc. 2016.

[14] Wang Y, Lin X, Pedram M. A game theoretic framework of sla-based resource allocation for competitive cloud service providers[Cll12014 Sixth Annual IEEE Green Technologies Conference. IEEE, 2014:37-43.

[15] Chinthagunta Mukundha, P.Gayatri, I.Suryaprabha, Load balance scheduling algorithm for serving of requests in cloud networks using software defined networks.International journal of Applied Engineering research 2016,Vol 11 No6,3910-3914.