A Load Balancing Game Approach for VM Provision Cloud Computing Based on Ant Colony Optimization

Khiet Thanh Bui[1(✉)], Tran Vu Pham[1], and Hung Cong Tran[2]

[1] Faculty of Computer Science and Engineering, Ho Chi Minh City University of Technology, Ho Chi Minh City, Vietnam
khietbt@tdmu.edu.vn
[2] Training and Science Technology Department, Posts and Telecoms Institute of Technology, Ho Chi Minh City, Vietnam

Abstract. The resource management on cloud computing is a major challenge. Resource management in cloud computing environment can be divided into two phases: resource provisioning and resource scheduling. In this paper, we propose VM provision solution ensure to balance the goals of the party stakeholders including service providers and customers based on game theory. The optimal or near optimal solution is approximated by meta-heuristic algorithm – Ant Colony Optimization (ACO) based on Nash equilibrium. In the experiments, the Ant System, Max-Min Ant System, Ant Colony System algorithm are applied to solve the game. The simulation results show how to use the coefficients to achieve load balancing in VM provision. These coefficients depend on objectives of cloud computing service providers.

Keywords: Load balancing · VM provision · Non-cooperative game · Ant Colony Optimization

1 Introduction

Infrastructure as a Service - IaaS cloud computing gives users such as network infrastructure, servers, CPU, memory, storage space as a virtual machine (VM) using server virtualization technology. Server virtualization technology allows to create multiple virtual machines on a physical machine (PM) and each VM is allocated in hardware resources as real machine with RAM, CPU, network card, hard drive, operating system and the individual applications. The resource management on cloud computing is a major challenge. Resource management in cloud computing environment can be divided into two phases: resource provisioning and resource scheduling. Resource provisioning phase determine resource requirements as well as quality of service for the customer which will be allocated somewhere in the system. Resource scheduling phase manage the life cycle of resource after it is allocated successfully. Customers and service providers often have different requirements and may conflict with each other. Service providers want to maximize profits by maximizing use of resources. However, maximum exploitation of resources may not satisfy customers

P. Cong Vinh et al. (Eds.): ICCASA 2016, LNICST 193, pp. 52–63, 2017.
DOI: 10.1007/978-3-319-56357-2_6

with performance and quality of service provided. To ensure quality of services, providers must extend the same or refuse new service requirements. Optimal resource provision is essential in the use of cloud computing resources especially IaaS. Optimization problems of this type are usually NP-Hard class or NP-Complete [1]. Solution to this problems are usually based on specific characteristics which apply algorithms such as exhaustive algorithm, deterministic algorithm [2] or algorithm meta-heuristic [3–5]. In experiments, almost deterministic algorithms are than better algorithms exhaustive. However, deterministic algorithms are ineffective in distributed data environment, thereby leading to inappropriate scheduling issues in large-scale environment [6]. Meanwhile, cloud computing environment data is distributed, requiring scalability, ability to meet high customer requirements to access VM provision problems by meta-heuristic is feasible. Although meta-heuristic algorithms can give near optimal results in acceptable time, in this study, we propose provision solution ensure to balance the goals of the party stakeholders including service providers and customers based on game theory. Then, the algorithm used in particular meta-heuristic is Ant Colony Optimization (ACO) to find solutions which is an optimal or near optimal VM provision based on Nash equilibrium. The remainder of this paper is organized as follows. Section 2 describes state or art of the model game to solve resource provision. A game load balancing VM provision model is described in Sect. 3. The main purpose of Sect. 4 describes the ACO algorithms for the VM provision in cloud computing. Performance evaluation of the proposed and simulation results is described in Sect. 5. Finally, Sect. 6 presents the conclusions.

2 Related Work

According Grosu et al., there are 3 types of models for load balancing in single issues distributed to job class system: global, cooperative, non-cooperative [7]. The article suggested using Nash Bargaining Solution to provide a Pareto optimal allocation that fair allocation for all the jobs. The Fairness index is always one using the NBS which means allocation is fair to all jobs. Besides, load balancing in heterogeneous distributed system towards user-modeled optimal is proposed non-cooperative game by Grosu et al. proposed in [8]. But this proposal is only applied in static load balancing model, nor the dynamic load balancing model. For the proposed non-cooperative load balancing game, Aote et al. consider the structure of the Nash equilibrium [9]. They define the load balancing problem and the scheme to overcome it by using new area called game theory. Based on this structure they derive a new distributed load balancing algorithm. Minarolli et al. proposed CPU allocation for VMs in the IaaS cloud based on QoS aspects and operating cost [10]. Resource management model includes 2 levels: local controller undertake CPU allocation for VM to achieve optimal at the PM locally and global controller manages the VMs and live migration to other physical machines for achieving maximum global utility system. But this article is only interested in CPU resources that have not mentioned memory, disk and network. Using migration technique, Yang et al., guarantee for the full balance of the global system [11]. Ye et al. proposed non-cooperative games Strategic model for both the load balancing server problem and virtual machine placement problem [12]. The load balancing server

problem is mapping a set of VMs which is described as a multi-dimensional vector to PM to achieve maximum load on a PM on any minimized dimension. VM placement problem is a set of VMs is assigned to the minimal number of PMs in which the load on each PM is in limited capacity. The VM and the PM are not identical in terms of capacity and configuration,… The VMs mapping PMs ensures using PMs resource efficiently. Efficiency issues are considered as the overload of the physical machine as using fewer machines for energy-saving materials [13]. This article suggests demand of forecasting algorithm based on resource using exponentially weighted moving average EWMA. Optimal changes in the use of resources on a physical machine to achieve the optimal across the system. This method can only achieve local optimal, not global optimization. This article only addresses allocation issues when required. When a request arrives, the service providers must decide whether to accept or not to meet the requirements of the system requirements of vendors that can handle this request in the external system - an affiliate vendor. To solve that problem, Tchernykh et al. modeled the problem towards energy-Efficient [14]. Algorithms scheduling algorithms are evaluated on the income provider and power consumption. To achieve the fairness between systems and customers in distributed systems, Siar et al. modeled the problem in non-cooperative game [15]. Using genetic algorithms and hybrid popularity algorithm is to find the optimal solution or the near optimal based on Nash equilibrium. In [16] Considering the demands of end users and service providers achieves multi-QoS indexes by calculating the load of each peer through quantitative analysis of costs, system and network. These are peer ratings, thanks to the weights determining whether peer matching the requirements of users while ensuring optimal goal of using resources to save money. Sui et al. proposed strategy and recoding Spectrum sharing on non-navigation-driven selection and Nash equilibrium cooperative game [17]. But in general cases, it is difficult to achieve all solutions. The best solution in the Nash game cannot describe the dynamic change of strategy players. Therefore, the Evolutionary game theory for network selection is proposed. [18] VM scheduling problem is solved by combining ant colony optimization algorithm and dynamic VM forecast scheduling (VM_DFS). Through the analysis of the historical using memory in each servers predicts the possibility of using the memory of the VM on the server in the future which is important as a basis for finding the optimal solution for scheduling based on ant colony optimization algorithm.

3 A Game Load Balancing VM Provision Model

3.1 VM Provision

In IaaS cloud, PMs can deploy VMs on itself based on virtualization technique. A VM requirements r (cpu, ram, disk) correspond to cpu, ram, the virtual machine's disk. Ensuring the efficient use of resources as well as the use of infrastructure services IaaS stability, allocates resources strategically in IaaS virtual machine reasonable. Maybe modeling scheduling problems on the cloud as a directed graph DAG (Directed acyclic Graph)) [19–21] G (V, E) where V is a set vertex represent tasks, E is the set of directed

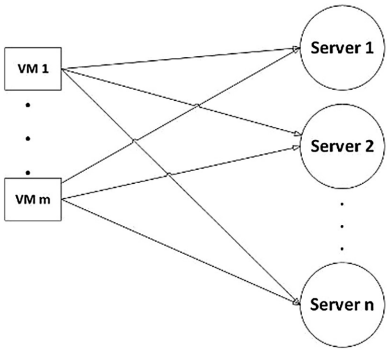

Fig. 1. VMs provision ability graph

edges represent dependency relationships between the vertices. In Fig. 1. Presents the ability of VM j is allocated on PM i.

3.2 A Game Load Balancing Model

To model this problem by game theory, we consider customers as the players in the VM provision game. To ensure the system is always efficient, well, how the system should maximize the use of PM resources evenly. To measure the efficiency of resource using a PM, using the following formula:

$$h_i = \frac{U_i}{T_i} \tag{1}$$

In which U_i is the resources that were used in the PM i is calculated using the formula:

$$U_i = \sqrt{c_i^{u^2} + r_i^{u^2} + d_i^{u^2}} \tag{2}$$

T_i is resources ith physical machine, is calculated as follows

$$T_i = \sqrt{c_i^2 + r_i^2 + d_i^2} \tag{3}$$

The load balancing system is measured by the following formula:

$$L = \frac{\sum_{i=1}^{n} (h_i - \bar{h})^2}{n} \tag{4}$$

The service providers avoid wasting the resources of the physical machine. When VM j is allocated on the host j, the waste of resources is calculated as follows:

$$W = \begin{cases} \sum_{i=1}^{k} A_i, \ k \in n\,hosts \setminus host\,j \\ 0, \ if\ vm\,j\,is\,not\,allocated \end{cases} \tag{5}$$

In which, A_i is ready to serve the resource requirements of the VM of PM i, it is calculated as follows:

$$A_i = \sqrt{c_i^{a^2} + r_i^{a^2} + d_i^{a^2}} \tag{6}$$

For parameter μ, λ ∈ [0.1] to perform trade-offs between load balancing and profit maximization. The payoff gives players the j required to serve VMs represented by a linear combination of L_j, W_j

$$F_j = \mu L_j + \lambda W_j \tag{7}$$

3.3 Problem Solving

Nash equilibrium is a strategy game in which no player can increase profits while other players have a fixed strategy. Meanwhile, if the strategy of the first player i's optimal strategy is denoted p_i^*, the optimal strategy of the other players is denoted by p_{-i}^*, the Nash equilibrium strategy p_i^* will comply with the conditions [22], as follows:

$$F_i\left(p_{-i}^*, p_i^*\right) \geq F_i\left(p_{-i}^*, p_i\right) \tag{8}$$

In multi-agent system environment, equilibrium can be unstable [23]. Also, it's hard to find Pareto-efficiency of Nash equilibrium. To solve this problem, most of the algorithm is based on the algorithm meta-heuristic. The plan assigns VMs to feasible PMs to find the optimal based on ant colonies algorithms. From feasible plan that is based on Nash equilibrium conditions will select the best plan. When no player can reach further payoff past estimated near optimal, it means that all players have selected their approximated Pareto optimal strategies [15]. If F_j^{itr} presents player's payoff j in iterator *itr* of the ant colony optimization algorithms, $F_j^{itr} - F_j^{itr-1}$ presents the improment of player j's payoff. Termination condition is the sum of square deviation of all players' payoff less than a small number ε, i.e.:

$$\sum_{i=1}^{n} \left(F_j^{itr} - F_j^{itr-1}\right)^2 < \varepsilon \tag{9}$$

4 Ant Colony System Algorithm for Allocation VM

Ant colony optimization algorithm is proposed based on experiments on ants. Due to the nature and chemical characteristics, every ant on the move always leaves a chemical trail called pheromone trail along the way and they often take the path with dense

smell. The pheromone trail is these chemicals evaporated over time. Enhancing the learning process has the effects of raising the efficiency of the algorithm in the process of the ants for finding the solutions. One of the first important thing in the application of the ACO algorithm is pheromone information. Here pheromone is likely a selected PM to allocate VM on demand, this ability depends on the current configuration and server heuristic information. Information heuristic will be recalculated after each allocation by the configuration information of the PM changes after each successfully allocated VM. The recalculation will enable more accurate heuristic information for next time allocation.

```
Algorithm : Ant Colony Optimization Meta-heuristic
```

```
While termination-condition not met do
 Initialization pheromone for host
 Initialization heuristic for host
  For each gamer
   For each request VM
    Calculate the probability of the valid Hosts
    Allocate VM Based on the probability of Hosts
    Update pheromone
   End For
  End For
 End While
```

4.1 Ant System Algorithm

The Ant System algorithm (AS) has two major phases: building local solutions and updating pheromone trail. A heuristic argument is said to be good when the index starts the initial pheromone value slightly higher than the number of pheromone can create in each turn building local solutions [24]. After each iteration of the algorithm, pheromone value are updated by the smell of all the ants that had built solution on its loop. Value τ_{ji} on edge (j,i) is calculated as follows:

$$\tau_{ji} \leftarrow (1-\rho).\tau_{ji} + \sum_{k=1}^{m} \Delta\tau_{ji}^{k} \tag{10}$$

In which, $0 < \rho <= 1$ is the rate of evaporation of the pheromone trail. Parameters evaporation avoids excessive accumulation streak pheromone and eliminates the inefficient PMs were selected earlier. $\Delta\tau_i^k$ presents quality smell of ants on the edges (j,i) on the graph are calculated as follows:

$$\Delta\tau_i^k = \begin{cases} \frac{Q}{L_k} \, if \; the \; ant \; k \; choose \; (j,i) \\ 0 \qquad otherwise \end{cases} \tag{11}$$

In which, Q is constant, L_k is the cost of the ant k through the edge (j, i).

For each VM requirement, the program will calculate to retrieve a valid PMs (which is eligible to allocate a VM) and the calculated probability of being selected for each PM. The probability of each valid PM selected to allocate VM is calculated using the formula:

$$p_{ji}^k = \begin{cases} \frac{\left(\left[\tau_{ji}\right]^{\alpha} \cdot \left[\eta_{ji}\right]^{\beta}\right)}{\sum_{c_{jl} \in N(s^p)} \left(\left[t_{jl}\right]^{\alpha} \cdot \left[\eta_{jl}\right]^{\beta}\right)} if \;\; c_{ji} \in \boldsymbol{N}(s^p) \\ 0 \quad otherwise \end{cases} \tag{12}$$

In which $\mathbf{N}(s^p)$ is a valid set of PMs can meet the required VM k. Edge (j,l) that has not been visited by ant k. The parameter α, β used to determine the effects of pheromone and information value η_i heuristic, heuristic information are calculated using the formula $\eta_i = h_i$

4.2 Max-Min Ant System Algorithm

MAX - MIN Ant System [25] called MMAS is an improved version of AS with four modifications: First, the only ant finds the best solution is updated pheromone, but this can lead to delay when looking for new and better solutions for the following ants tend to move in the directions of high pheromone concentrations (usually in the direction of the ants before). To avoid this, a change is proposed to create limited access markings odor 2: max and min. Pheromone value is updated as follows:

$$\tau_{ji} = \left[(1-\rho).\tau_{ji} + \Delta\tau_{ji}^{best}\right]_{\tau_{min}}^{\tau_{max}} \tag{13}$$

In which, the value τ_{max} and τ_{min} marginal value of pheromone, with operator

$$[x]_b^a = \begin{cases} a \, if \, x > a, \\ b \, if \, x < b, \\ x \; otherwise \end{cases} \tag{14}$$

$$\Delta\tau_{ji}^{best} = \begin{cases} \frac{1}{L_{best}} if \; (j,i) is \; used \\ 0 \quad otherwise \end{cases} \tag{15}$$

In which, L_{best} is the heuristic of hosts which the best ants choose.

The pheromone concentrations are initialized by the value of the upper, which will help to enhance the search for better solutions when searching. If the case could not find a better solution after several attempts, the pheromone concentration trail will be reset.

4.3 Ant Colony System Algorithm

Ant Colony System Algorithm (ACS) [26] is proposed by Dorigo and Gambardella. First, the ability to search the gradual increase is expected to be better than the algorithm AS using the action selection rules. Second, the process of evaporation and distributed pheromone occur only when the ant is choosing the best solution. Finally, every time is selected to be one good solution and distribution, it also reduces the pheromone trail of surrounding solution's pheromone concentration.

Probability selected PM to allocate VM requirements calculated under the pseudorandom proportional: the probability that an ant selects an edge (j,i) depends on the random variable normal distribution $q \in [0, 1]$,

$$p = \begin{cases} argmax_{c_{il} \in N(s^p)} \left\{ \tau_{il} \eta_{il}^{\beta} \right\}, if\ q \leq q_0 \\ p_{ji}^{k}, \quad if\ q > q_0 \end{cases} \tag{16}$$

Parameters q_0 will help choose the best PM in the iteration k by using heuristic information and pheromone concentration of PM. Using variable q_0, algorithm allows selection between PM – the best current configuration and looking for a different approach – a PM with more appropriate configuration.

In this algorithm, only the PM is determined to be the best in the iteration shall be updated pheromone. This can greatly affect the algorithm performance, the complexity of the algorithm in the function updates pheromone trail will be reduced from O (n^2) to O (n) (because of the need to update PM pheromone in each level found only 1). Pheromone value is updated as follows:

$$\tau_{ji} \leftarrow \begin{cases} (1-\rho)\tau_{ji} + \rho \Delta \tau_{ji}\ \hat{ne'u}\ (j,i)\ is\ the\ best \\ \tau_{ji} \quad otherwise \end{cases} \tag{17}$$

Like MMAS algorithm, L_{best} is the heuristic of hosts which the best ants choose.

5 Simulation Results

In this paper, we are concerned with the problems of the load balancing and resource extraction. With optimal ants algorithm classes, the results depend on the parameters $\varepsilon, \alpha, \beta$. Thus, in the experiments below, we find the appropriate parameters for the algorithms as well as the allocation of resources for customer VMs through load balancing levels of the system in the formula (4) and the level of resource wasting system resources by the formula (5).

In the Fig. 2 changing ε from 0.03 to 0.1, we can see ACS algorithm with a few numbers of iteration is stable. When increasing epsilon, MMAS and AS algorithms have reduced the number of iteration and are nearly equal to the number of iterations of the ACS. The iterations of MMAS is higher than AS and ACS. This shows that the algorithm MMAS has richer solutions.

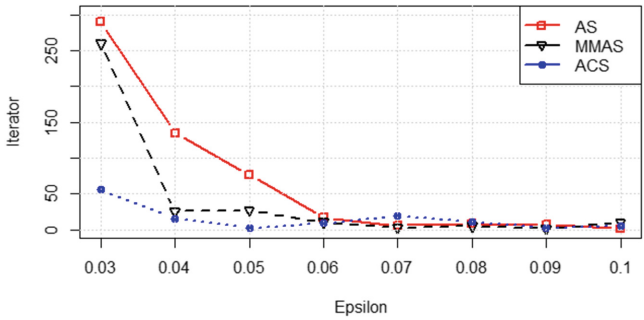

Fig. 2. Iteration of the algorithm with ε

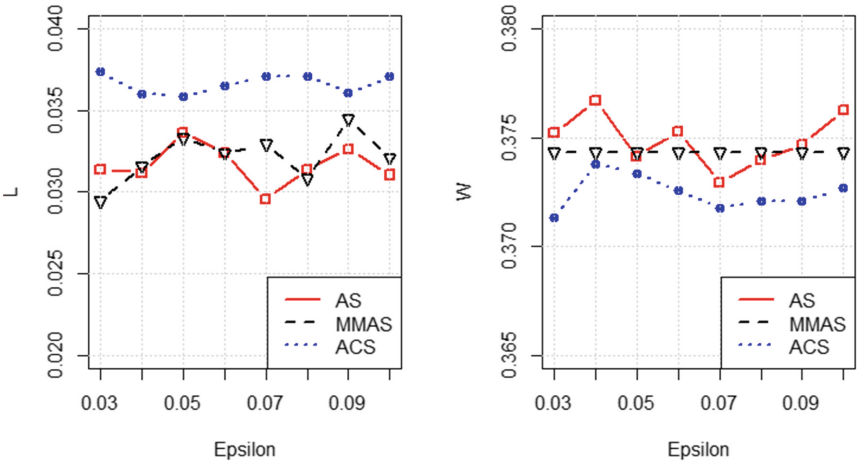

Fig. 3. Load balancing level and waste resource level with ε

In Fig. 3 when ε increases, the level load balancing and waste resource tends to increase. Therefore, within the limits of the paper we choose epsilon = 0.05 level for the next experiment. Choose other parameters $\varepsilon = 0.05$; $\mu = \lambda = 0.5$; $\rho = 0.2$; $p_0 = 0.9$. Let the number of clients from 10 to 70. Change α, β, then the load balancing level and waste resource level presents following (Figs. 4 and 5):

In general view, the load balancing level of all of algorithms has increased the propensity. We consider the load balancing level of all the algorithms. The ACS algorithm is more stable than AS and MMAS but its value is higher than AS and MMAS. The fluctuation band of MMAS is higher. When the α is larger than the β, all of algorithm has the same propensity.

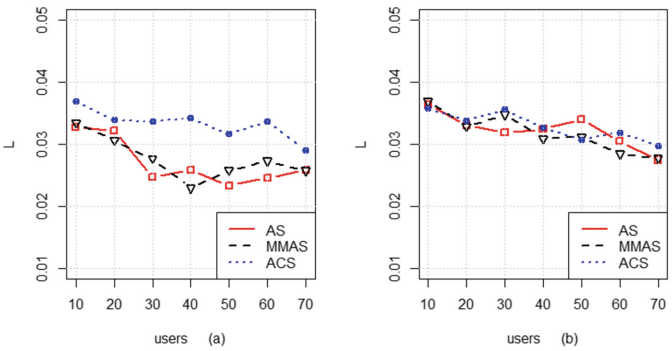

Fig. 4. (a) $\alpha = 0.1$; $\beta = 0.9$; (b). $\alpha = 0.9$, $\beta = 0.1$.

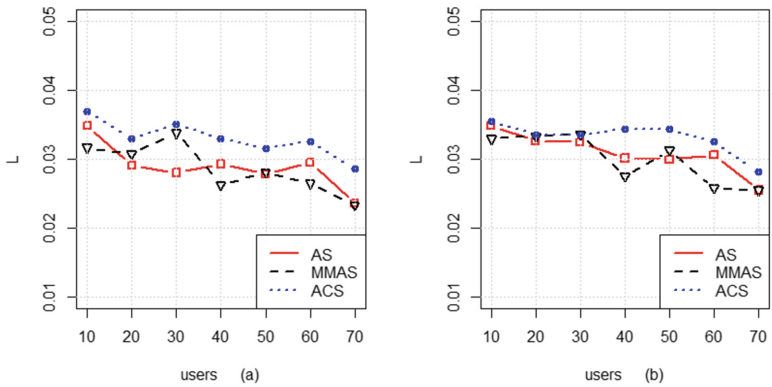

Fig. 5. (a) $\alpha = 0.3$; $\beta = 0.7$; (b) $\alpha = 0.7$; $\beta = 0.3$

6 Conclusions

In this paper, we use the non-cooperative game for the gamers in the VM provision to achieve load balancing. The game's payoff calculated by combining the two parameters load balancing and the waste resources. The load balancing parameter helps to distribute VMs to PMs based on current state of PMs. The waste resource helps service providers to achieve optimal profits. We propose the load balancing model take care both of customer and service provider are by using combination two parameters. The optimal solution is approximated by ant colony optimization based on Nash equilibrium. The experiments show how to use the coefficients to achieve load balancing in VM provision. These coefficients depend on objectives of the cloud computing service provider. In the next time, we study another optimization algorithms for this problem as well as study how to use the coefficients to achieve optimal or near optimal solution.

Acknowledgement. This work is supported by the Thu Dau Mot University's research program in 2016.

References

1. Gary, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-completeness. WH Freeman and Company, New York (1979)
2. Morton, T., Pentico, D.W.: Heuristic Scheduling Systems: With Applications to Production Systems and Project Management. Wiley, New York (1993)
3. Van Laarhoven, P.J., Aarts, E.H., Lenstra, J.K.: Job shop scheduling by simulated annealing. Oper. Res. **40**, 113–125 (1992)
4. Colorni, A., Dorigo, M., Maniezzo, V., Trubian, M.: Ant system for job-shop scheduling. Belg. J. Oper. Res. Stat. Comput. Sci. **34**, 39–53 (1994)
5. Ghumman, N.S., Kaur, R.: Dynamic combination of improved max-min and ant colony algorithm for load balancing in cloud system. In: 2015 6th International Conference on Computing, Communication and Networking Technologies (ICCCNT), pp. 1–5. IEEE (2015)
6. Tsai, C.-W., Rodrigues, J.J.: Metaheuristic scheduling for cloud: a survey. IEEE Syst. J. **8**, 279–291 (2014)
7. Grosu, D., Chronopoulos, A.T., Leung, M.-Y.: Load balancing in distributed systems: an approach using cooperative games. In: Proceedings of the IEEE-IEE Vehicle Navigation and Information Systems Conference 1993, p. 10. IEEE (1993)
8. Grosu, D., Chronopoulos, A.T.: Noncooperative load balancing in distributed systems. J. Parallel Distrib. Comput. **65**, 1022–1034 (2005)
9. Aote, S.S., Kharat, M.: A game-theoretic model for dynamic load balancing in distributed systems. In: Proceedings of the International Conference on Advances in Computing, Communication and Control, pp. 235–238. ACM (2009)
10. Minarolli, D., Freisleben, B.: Utility-based resource allocation for virtual machines in cloud computing. In: 2011 IEEE Symposium on Computers and Communications (ISCC), pp. 410–417. IEEE (2011)
11. Yang, C.-T., Cheng, H.-Y., Huang, K.-L.: A dynamic resource allocation model for virtual machine management on cloud. In: Kim, T.-h., Adeli, H., Cho, H.-s., Gervasi, O., Yau, Stephen, S., Kang, B.-H., Villalba, J.G. (eds.) GDC 2011. CCIS, vol. 261, pp. 581–590. Springer, Heidelberg (2011). doi:10.1007/978-3-642-27180-9_70
12. Ye, D., Chen, J.: Non-cooperative games on multidimensional resource allocation. Future Gener. Comput. Syst. **29**, 1345–1352 (2013)
13. Xiao, Z., Song, W., Chen, Q.: Dynamic resource allocation using virtual machines for cloud computing environment. IEEE Trans. Parallel Distrib. Syst. **24**, 1107–1117 (2013)
14. Tchernykh, A., Lozano, L., Bouvry, P., Pecero, J.E., Schwiegelshohn, U., Nesmachnow, S.: Energy-aware online scheduling: ensuring quality of service for IaaS clouds. In: 2014 International Conference on High Performance Computing & Simulation (HPCS), pp. 911–918. IEEE (2014)
15. Siar, H., Kiani, K., Chronopoulos, A.T.: An effective game theoretic static load balancing applied to distributed computing. Cluster Comput. **18**, 1609–1623 (2015)
16. Liu, L., Mei, H., Xie, B.: Towards a multi-QoS human-centric cloud computing load balance resource allocation method. J. Supercomputing **72**, 2488–2501 (2015)
17. Sui, N., Zhang, D., Zhong, W., Wu, L., Zhang, Z.: Evolutionary game theory based network selection for constrained heterogeneous networks. In: 2015 2nd International Conference on Information Science and Control Engineering (ICISCE), pp. 738–742. IEEE (2015)
18. Seddigh, M., Taheri, H., Sharifian, S.: Dynamic prediction scheduling for virtual machine placement via ant colony optimization. In: 2015 Signal Processing and Intelligent Systems Conference (SPIS), pp. 104–108. IEEE (2015)

19. Li, J., Qiu, M., Ming, Z., Quan, G., Qin, X., Gu, Z.: Online optimization for scheduling preemptable tasks on IaaS cloud systems. J. Parallel Distrib. Comput. **72**, 666–677 (2012)
20. Rahman, M., Li, X., Palit, H.: Hybrid heuristic for scheduling data analytics workflow applications in hybrid cloud environment. In: 2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), pp. 966–974. IEEE (2011)
21. Saovapakhiran, B., Michailidis, G., Devetsikiotis, M.: Aggregated-DAG scheduling for job flow maximization in heterogeneous cloud computing. In: 2011 IEEE Global Telecommunications Conference (GLOBECOM 2011), pp. 1–6. IEEE (2011)
22. Osborne, M.J., Rubinstein, A.: A Course in Game Theory. MIT Press, Cambridge (1994)
23. Pendharkar, P.C.: Game theoretical applications for multi-agent systems. Expert Syst. Appl. **39**, 273–279 (2012)
24. Dorigo, M., Maniezzo, V., Colorni, A.: Ant system: optimization by a colony of cooperating agents. IEEE Trans. Syst. Man Cybern. Part B (Cybernetics) **26**, 29–41 (1996)
25. Stützle, T., Hoos, H.H.: MAX–MIN ant system. Future Gener. Comput. Syst. **16**, 889–914 (2000)
26. Dorigo, M., Gambardella, L.M.: Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans. Evol. Comput. **1**, 53–66 (1997)