Resource Allocation Using Game Theoretic Model in Cloud Computing

Sushmitha. N.R (BCA, 6th Sem) - 1st
Dept. of B.C.A,
Jain University,
Bangalore, India.
sushmitha2mail@gmail.com

Vikash Kumar - 2nd
Dept. of IT,
Inurture Education Solutions Pvt. Ltd.
Bangalore, India.
Vikash.k@inurture.co.in

Abstract-Cloud computing provides computational resources as a highly scalable service in a pay-as-you-go model and implements high performance computing in a distributed way. This paper proposes a resource allocation algorithm based on game theory for multi resource environment. Each physical server providing resources is treated as a game player and knows the utility information of other players. To achieve the fair allocation among users while keeping a high resource utilization level, we design a 'fairness utilization trade-off utility function'. This paper focuses also on trying to maximize the minimum consumption among these multiple resources and lowering the uneven consumption of different resources [1].

The major contributions of this paper as follows:

1. A cloud resource management system is designed to provide on demand resources in time.

2. A game theoretic resource allocation algorithm is proposed to get an optimal resource allocation decision, which makes sure thefairness of multiple resources sharing among separated users and reduces the resource fragments to increase the efficiency.

3. The multi resource allocation problem on virtual machines level is designed by trading off fairness and resource utilization.

Index Terms - Fairness, game, Nash equilibrium, players, strategy, VM

1. INTRODUCTION

Game theory has been applied to solve resource allocation problem in cloud computing. Ye and Chen study non cooperative games for the cloud balancing and virtual machines placement problems. They focus on the existence of NashEquilibrium and little about the solution of an optimal allocation strategy. Our work focuses on the allocation problem in multi resources environment.

1.1 RESOURCE MANAGEMENT SYSTEM

We are interested in providing fair and effective resource allocation mechanism on a distributed and complex cloud system thus;a resource management system is necessary to centralized control and coordinates the physical resources. This resource management system has four components which includes register center (RC), cloud environment monitor (CEM), infrastructural management (IM) and control center (CC).[1]

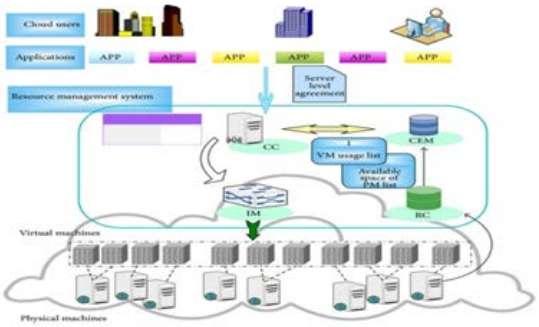

Figure 1: A framework of cloud resource management system.

CEM is monitoring the statuses and resource consumptions for physical server started to join the cloud, the information like MAC address, IP address will be registered RC. When a user sends a

service request to cloud, the requirements of resources in this request will be received by control center. It makes an intelligent resource allocation decision based on the information collected by CEM. The allocation decision is executed by IM to manage the physical servers and the place of VMs. In our resource management system, the allocation of resources is in a time slotted paradigm. The dynamically arriving user request of current time slot are recorded and will be served for resource allocation at the start of next time slot.

1.2 GAME THEORETIC RESOURCE ALLOCATION

Each user in a cloud asks for a type of VM to run its job. The execution of a job involves multidimensional resources and the resource requirements differ from job to job. For example, a data mining job needs high capacity of disks to store a large amount of data while a calculating job might need more CPU than disk to get a result. In order to support elastic multi resource consumption, we propose a fairness-utilization tradeoff algorithm (FUGA), which makes an optimal tradeoff between fairness and efficiency

2. FAIR ALLOCATIONS

In this paper, fair allocation problem is considered for multiple types of resources. For a single type of resource, fair allocation means each user has equal share of resources. However, in multi resource environment, since users have heterogeneous requirements for different types of resources, resources should be assigned to users in proportion to their requirements. Each user has a maximum share fraction of total capacity among different resources which is called dominant share. The major goal of fair allocation considered in our work is to equalize the dominant share of each user. [1]

To achieve fair allocation it should satisfy three widely used properties:

1. **Sharing incentive**: Sharing incentive means the amount of resource user should receive is at least as much as

Simply splitting the total resources equally

2. **Envy freeness**: Envy freeness is the property that no user prefers to the allocation of another user

3. **Pareto efficient**: It should be impossible to increase the resource amount of a user without decreasing the allocation of another user.

The fairness of multiple resources sharing is measured by extending the dominant resource fairness (DRF) mechanism that Ghods put forward at 2011. In words, to mathematically gauge the fairness of a resource allocation mechanism, the PRF is set to be the benchmark of fair allocation. Each allocation decision may have a deviation contrast to the fair allocation called fairness variance.

Given a resource requirement matrix R and the summation of the total resources for all physical servers.

$C= (\sum_m C_1^{(m)}, \sum_m C_2^{(m)} \ldots \sum_m C_j^{(m)} \ldots \sum_m C_k^{(n)})$

The first step is to normalize the requirement matrix. The normalized matrix is denoted by Ψ [2]

Secondly, as mentioned before, the dominant share of a user is the largest fraction of any kinds of resources allocated to that user. **Let $d_{ij} = \Psi_{ij} / (\max_j \Psi_{ij})$** be the normalized demands, and

$\lambda = 1 / (\max_j \sum_i d_{ij})$ is the dominant share.

3. THE FAIRNESS UTILIZATION TRADE-OFF GAME ALGORITHM

Resource allocation game:

Game theory is a mathematical study of strategy which attempts to determine the interactions among all game players to ensure the best outcomes for themselves. A game consists of three factors, that is, a set of players, all the possible strategies each player will choose, and the specified utilities of players associated with the strategy performed by every player. At each step, players choose one of their strategies and get a utility in return. Each player of a game tries to maximize its own utility by choosing the most profitable strategy against other players' choices. Nash Equilibrium is a central notion of game theory which means in this situation no player can get more utilities by changing its strategy.

A specification of a game is an extensive game which provides the sequencing of all players' possible strategies and their decision points. A finite extensive game with perfect information has a finite set of players, and each player knows the information of other players' strategies and all possible utilities. A sub game perfect Nash Equilibrium (SPNE) is a solution such that players'

strategies constitute a Nash Equilibrium in every sub game of an original game.

In our work, the resource allocation problem is modeled as a finite extensive game with perfect information. Physical servers with idle resources are modeled as the selfish players and each player has a limited number of possible allocation matrices.

The following symbols are introduced to define the resource allocation game

A resource allocation game is represented as a four-tuple vector G= (P, R, A, U).

(i) P is the players in the allocation game.

(ii) R refers to the resource requirements matrix of users.

(iii) A are the sets of players' strategies.

(iv) U is the utility function of game players.

At decision moment, CC gets the resources consumption information of each physical server in data center from CEM. P is represented for the set of physical servers with idle resources and each server is associated with a capacity vector. All the users' requests for cloud resources CC receives during last time-slot are analyzed and transformed to the resource requirement matrix R.

For a physical server, there are a variety of possible combinations to be fulfilled by different types of VMs without exceeding the capacity A combination of physical server m can be denoted as $com_x(m)=[c_{x1}, c_{x2}, ...,c_{xs}]$.

For instance, the cloud users ask for three VM types, r1, r_2, r_3, and corresponding to vectors (2, 4, 20), (1, 1, 10), and (2, 2, 10) and physical sever 1 has (4, 8, 40) capacity of spare resources <1, 1, 0>, means one VM of type r_1 and one VM of type r_2 can be created on physical server m.

In this resource allocation game, the physical servers with idle resources are game players, and they are individual rationality to maximize their own utilities. Based on the discussions in previous, the design of the utility function has a crucial impact on players' choices and the result of the game. In our allocation model, one global objective of this allocation game is to share resources impartiality [1]. Furthermore, based on the efficient principle each individual player tries to minimize their resource wastage, that is, they prefer to choose those combinations with high utilization. To exploit fair resource sharing and also take the maximization of resource utilization rate into account, a fairness-utilization tradeoff utility function is designed as follows:

$U^{(m)}(A) = sgn(1-\alpha).\ v(A) - Ske(m)$ [3]

α is a coefficient to affect the weights of fairness and utilization. V (A) is the fairness variance and is the ske (m) is the skewness which reflects the unevenness for the utilization of different resources. The less fairness variance an allocation decision A gets, the more utilities players gain. Similarly, each physical server prefers to choose the combination with less skewness to optimal its own utility.

Each player of this game aims to choose a strategy to maximize its own utility so that the goal of a resource allocation game would be naturally considered as the following optimization problem:

Maximize $U^{(m)}(A)$

$$\text{Subject to } \sum_i \sum_m a_{ij}{}^{(m)} \leq C_j$$

$a_{ij}{}^{(m)} \geq 0$

$A^* = \{ A^{(1)*}, A^{(2)*}, ...,A^{(m)*}, ...A^{(p)*} \}$

is the Nash equilibrium of a resource allocation game which means for all m, $U(A^1, A^2,,.....A^m.....A^p) > U(A^1, A^2,.....A^{m'} .....A^p)$.[4]

Input: $\{C^{(m)}\}$, R

Output: A^*

1. Start
   Initialization: combinlists, SelectedServerLists, selection[p+1]
2. Step-1 : Phase before combination
3. //each and every physical server with idle resource will be a player
4. p←{1,….m,……p}
5. for each physical server m do
6. listing the possible combinations of this server to be fulfilled by different types of VMs without exceeding the capacity in the $combinList_m$
7. combinLists.add($combinList_m$)
8. end of for
9. Step-2 : Strategies set for each player
10. For each physical server m do
11. Pick the top $\wp$ of combinations $O^{(m)}$ ={com_1,….com_2,….com_n} and calculate min ($O^{(m)}$)

12. //each $com_x(m)$ can be represented as an allocation matrix $A_x^{(m)}$
13. End for
14. Step-3: generating extension-form game tree
15. The original array [min $(O^{(1)})$, ……min$(O^{(p)})$] is re-arranged in a non-decreasing order with indices[i_1…..i_p] such that min$(O^{(i_1)})$ ≤……≤ min $\left(O^{(i_p)}\right)$
16. The game players take action as an order of [i_1…….i_p]
17. Step-4 : Find the SPNE for a game G
18. For each strategy $A_x^{(i_{p-1})}$, $A_y^{(i_p)}$ of physical server i_{p-1}, i_p do
19. Calculate the utility pair$\left(U^{i_{(p-1)}}[x][y], U^{i_p}[x][y]\right)$
20. End for
21. Max[x]←argMax$_x U^{i_{(p-1)}}$[x]{max[x]}
22. Selection[$i_{(p-1)}$]←argMax$_x U^{(i_{p-1})}$[x]{max[x]}
23. Selection[i_{p-1}]←argMax$_x$ (max[x])
24. Add $i_{(p-1)}$, i_p to the selectedServerList
25. For each physical server m from i_{p-1} to 1 do
26. Add the total amount of resources $\wp$ for physical servers in selectedServerList
27. For each strategy $A_{x^{(m)}}$ of physical server m do
28. Calculate the ske(m) if $A_{x^{(m)}}$ is chosen
29. Add up the total allocated resources $\wp = A^m + \wp$
30. Calculate the v(A)
31. Utility calculation (ske(m), v(A))
32. End for
33. Selection[m]←argMax$_x(U^m\ [x])$
34. Add server m to selectedServerList
35. End for
36. The best strategy of each player m in selection[m] can be represented as an allocation matrix A^m , and A^* = $\{A^{(1)}……A^{(m)}…..A^{\{p)}\}$
37. Stop [1]

4. PERFORMANCE EVALUATION AND COMPARISON

This section presents a comprehensive evaluation of the resource allocation algorithm proposed in the previous section. The evaluation of fairness is done through a prototype implementation of our FUGA algorithm running on an 8-node cluster first [4]. And then the conduct of Google Trace-driven simulations shows that FUGA is efficient in improving the resource utilization by contrast with the First-Fit Algorithm and the management mechanism of Google cluster

5. EXPERIMENTAL ENVIRONMENT

32 The experiments to evaluate the performance of fair allocation were done on a small scale cluster with 8 physical nodes which consist of a Dell PowerEdge R910 with two CPUs (Xeon E7-4820 2 GHz 8cores), GB memory, and 300 GB disk storage, three Dell Optiplex9010 with one CPU (i7-3770 3.40 GHz 4cores), 8 GB memory, and 500 GB disk storage, and four Dell Optiplex745 with two CPUs (6600 2.4 GHz 2cores), 4 GB memory, and 200 GB disk storage. Three kinds of resource considered in this experiment include CPU, memory, and disk storage. The simulations were run on a Dell Optiplex9010 with JDK 1.7. To reduce the complexity of simulations, the following assumptions are made: (1) two kinds of resources (i.e., CPU and memory) are considered in our simulations. (2) Each job request submitted by a user indicates the predicted maximum consumption of different resources and will be handled by a cluster of VMs with the same type. (3) The total amount of resources provided for each time slot is previously estimated by cloud provider.

6. CONCLUSION

In this paper, we have investigated the resource allocation problem in cloud computing. We consider multiple types of resources like CPU, memory, and storage on virtual machine level to propose an allocation algorithm called FUGA. The algorithm supports not only fair resource allocation for users, but also efficient resource utilization for each physical server. The resource allocation problem is modeled as a finite extensive game with perfect information and the FUGA algorithm results in a Nash equilibrium decision.
Some experiments and simulations are conducted to evaluate the performance of FUGA by comparing to other related works. The results show that the proposed FUGA can achieve better performance in fair allocation than Hadoop scheduler. FUGA can also guarantee more efficient

resource allocation rather than the first fit algorithm and the allocation mechanism in Google cluster by setting the proper parameters for the fairness and utilization tradeoff.

Future work could usefully study the fairness-utilization tradeoff when jobs have machine preferences. Another direction involves considering the allocation problem under the job priority situation. Moreover, we plan to investigate how to use this game theoretic resource allocation into a federated environment with multiple resource providers.

7. REFERENCES

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms:vision, hype, and reality for delivering computing as the 5th utility," Future Generation Computer Systems, vol. 25, no. 6, pp. 599–616, 2009. View at Publisher · View at Google Scholar · View at Scopus.

[2] J. O. Gutierrez-Garcia and K. M. Sim, "GA-based cloud resource estimation for agent-based execution of bag-of-tasks applications," Information Systems Frontiers, vol. 14, no. 4, pp. 925–951, 2012. View at Publisher · View at Google Scholar · View at Scopus.

[3] J. Bi, Z. Zhu, R. Tian, and Q. Wang, "Dynamic provisioning modeling for virtualized multi-tier applications in cloud data center," in Proceedings of the 3rd IEEE International Conference on Cloud Computing (CLOUD '10), pp. 370–377, July 2010. View at Publisher · View at Google Scholar · View at Scopus.

[4] D. C. Vanderster, N. J. Dimopoulos, R. Parra-Hernandez, and R. J. Sobie, "Resource allocation on computational grids using a utility model and the knapsack problem," Future Generation Computer Systems, vol. 25, no. 1, pp. 35–50, 2009. View at Publisher · View at Google Scholar · View at Scopus.

[5] L. Guijarro, V. Pla, J. R. Vidal, and J. Martinez-Bauset, "Entry, competition, and regulation in cognitive radio scenarios: a simple game theory model," Mathematical Problems in Engineering, vol. 2012, Article ID 620972, 13 pages, 2012. View at Publisher · View at Google Scholar · View at Zentralblatt MATH.

[6] Iqbal and A. H. Toor, "Quantum mechanics gives stability to a Nash equilibrium," Physical Review A, vol. 65, Article ID 022306, 5 pages, 2002. View at Publisher · View at Google Scholar · View at MathSciNet.

[7] P. J. Reny, "Backward induction, normal form perfection and explicable equilibria," Econometrica, vol. 60, no. 3, pp. 627–649, 1992. View at Publisher View at Google Scholar · View at Zentralblatt MATH · View at MathSciNet.

[8] F. Schuhmacher, "Proper rationalizability and backward induction," International Journal of Game Theory, vol. 28, no. 4, pp. 599–615, 1999. View at Publisher · View at Google Scholar · View at Zentralblatt MATH. View at MathSciNet.