Deadlock Avoidance for Resource Allocation Model V VM-out-of-N PM

Ha Huy Cuong Nguyen[1(✉)], Hoang Dung Tran[2], Van Thang Doan[3], and Vu Thi Phuong Anh[4]

[1] Department of Information Technology, Quangnam University, Tam Ky, Vietnam
nguyenhahuycuong@gmail.com
[2] College of Food Industry, Danang City, Vietnam
dungdnt@gmail.com
[3] Ho Chi Minh City Industry and Trade College, Ho Chi Minh City, Vietnam
vanthangdn@gmail.com
[4] Quangnam University, Tam Ky, Vietnam
vuphuonganhdbqh@gmail.com

Abstract. This paper, presents an deadlock avoidance for model V VM-out-of-N PM. Algorithm used to reschedule the policies of resource supply for resource allocation on heterogeneous distributed platform. In the current scenario, deadlock avoidance for model V VM-out-of-N PM algorithm using Two - Way search method has created the problem of taking higher time complexity of O(m*(n − 1)/2 + 2e) where e is the number of edges, for m processes at n sites. This paper proposes the algorithms for allocating multiple resources to competing services running in virtual machines on a heterogeneous distributed platform. We have implemented and performed our algorithm proposed by using *CloudSim* simulator. The experiments results show that our algorithm can quickly avoid deadlock and then resolve the situation of approximately orders of magnitude in practical cases.

Keywords: Cloud computing · Resource allocation · Heterogeneous distributed platforms · Deadlock avoidance

1 Introduction

In the past, grid computing and batch scheduling have both been commonly used for large scale computation. Cloud computing presents a different resource allocation paradigm than either grids or batch schedulers [1,3,4]. Infrastructure as a Service (IaaS) is the concept of resource allocation hardware as a service. It allocation the required hardware resources offers CPU, RAM, HDD and software resources. Infrastructure as a Service is the Virtual Machine (VM) in heterogeneous. The introduction of heterogeneity allows clouds to be competitive with traditional distributed computing systems, which often consist of various types of architecture as well. Recently, reports have appeared that many of the studies provide cloud computing resources, the majority of this research to deal with

P. Cong Vinh et al. (Eds.): ICCASA 2016, LNICST 193, pp. 172–182, 2017.
DOI: 10.1007/978-3-319-56357-2_18

variability in resource capacity for infrastructure and application performance in the cloud. In this paper, we develop a method to avoid a deadlock occurs in the process of providing resources in class infrastructure as a service. Our rating indicates that the deadlock avoidance method using Two-Way search algorithm may improve the effectiveness and efficiency of resource allocation for heterogeneous distributed platforms.

In this work, we propose a deadlock avoidance algorithm, to avoid deadlock in resources allocation for model V VM-out-of-N PM. More specifically, our contributions are as follows:

1. Based on the resource model provides P-out-of-Q. We develop the resource model provides multiple virtual machines on multiple physical machines scattered ***V*** VM-out-of-***N*** PM.
2. We provide an algorithmic to request and to avoid deadlock in resource allocation for model V VM-out-of-N PM. This algorithm is, in fact, more generally, even for heterogeneous distributed platforms, and only allows allocating minimal resources to meet QoS (*Quality of Service*) arbitrary force.
3. We have studied the effects not effective in resources allocation, predict failures may occur in the system heterogeneous and suggest different approaches to mitigate this problem, followed by set out a strategy of automation in providing resources.

The work is organized as follows: Section 2 describes existing models; Section 3 provides the related works; Section 4 presents algorithm; Section 5 experiments and results and Sect. 6 presents our conclusions and suggestions for future work finally.

In this paper, we consider the proposed model resources allocation ***V*** VM-out-of-***N*** PM. From this model, we propose two algorithms to solve the problem. That algorithm resource requirements and algorithms avoid deadlock in resource allocation.

2 System model resource allocation in heterogeneous distributed platforms

2.1 The V VM-out-of-*N* PM model

A heterogeneous distributed platforms are composed of a set of an asynchronous processes (p_1, p_2, ...,p_n) that communicates by message passing over the communication network [4]. Based on the basic work of the authors Kshemkalyani-Singhal, and other works such as Menasce-Muntz, Gligor - Shattuck, Ho - Ramamoorthy, Obermarck, Chandy, and Choudhary [10]. They have the same opinions that the requested resource model of distributed system is divided into five model resource requirements. It's simple resource models, resource models OR, AND resource models, models AND/OR, and model resource requirements P-out-of-Q. Through this model, the researchers have discovered a technical proposal deadlock corresponding to each model. In this work, we use model

P-out-of-Q as a prerequisite for developing research models provide resources in the cloud. The $\boldsymbol{V}$ VM-out-of-$\boldsymbol{N}$ PM problem depicts on-demand resource allocation to $\boldsymbol{V}$ VMS residing in $\boldsymbol{N}$ servers, where each VM may use resources in more than one server concurrently. Thus, we model it to guide the design of algorithm avoid deadlock in resource allocation among VMs each of which may use the resource in various servers concurrently.

E_{ijt} is the amount of resources allocated to VM_{ij} at time t, where

$$\sum_{i}^{N} E_{ij} = \sum_{i}^{N} (A_{ij} + \sum_{i=1}^{V} C_{\mathrm{ij}}) \tag{1}$$

E_{ijt} obeys the rules as follows:

$$\begin{aligned} &E \geq \sum_{i=1}^{N} \sum_{j=1}^{V_N} E_{\mathrm{ij}t} \\ &E_{\mathrm{ij}t} \geq C_{\mathrm{ij}} \geq 0 (i = 1, ..., N; j = 1, ..., V_N) \end{aligned} \tag{2}$$

The resource allocation problem is how to control the resource allocation to VMs with the goal of minimizing the function F_t, giving the limited resources. We get the following formulation:

$$\begin{aligned} &F_t = min \sum_{i=1}^{N} \sum_{i=1}^{V_i} \frac{f_{\mathrm{ij}}(EN_{\mathrm{ij}t}, \sum_{x=1}^{V} EO_{\mathrm{ij}t}^{x}, D_{ijt})}{\Phi_{ij}} \times SP_{ij} \\ &\begin{cases} \sum_{i=1}^{N} \sum_{i=1}^{V_i} E_{ijt} \leq E \\ E_{\mathrm{ij}t} \geq C_{\mathrm{ij}} \ \ (i = 1, 2, ..., V; j = 1, 2, ..., N) \\ \sum_{i=1}^{V_i} EN_{\mathrm{ij}t} + \sum_{j=1}^{V_i} EO_{\mathrm{ij}t}^{i} \leq E_i \\ E_{it} \geq C_{ij} \ \ (i = 1, 2, ..., V; j = 1, 2, ..., N). \end{cases} \end{aligned} \tag{3}$$

We can use methods to avoid deadlock to solve optimal resource model provides $\boldsymbol{V}$ VM-out-of- $\boldsymbol{N}$ PM. Our algorithm is based on wait-for graphs (WFG) algorithm is presented in Sect. 4.

2.2 Problem with Definitions

The clustering is the subdivision of graph node set into groups. It partitions the graph into a smaller subdivision of connected sub graph called clusters. The techniques use the tree data structure to store the information about the graph edges and nodes [10].

In heterogeneous distributed platforms, the state of the system can be modeled by a directed graph, called a wait for graph (WFG). In a WFG, nodes are processes and there is a directed edge from node P_1 to node P_2 if P_1 is blocked and is waiting for P_2 to release some resources. A system is deadlocked if and only if there exists a directed cycle or knot in the WFG [10].

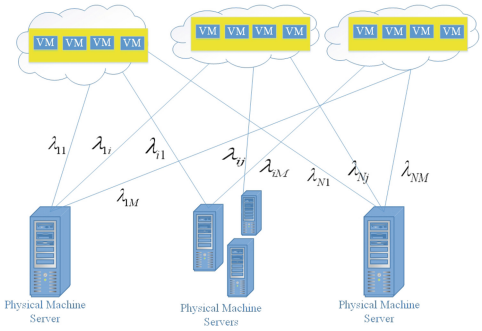

Fig. 1. A simple model ***V*** VM-out-of-***N*** PM

An allocation of resources to a virtual machine specifies the maximum amount of each individual element of each resource type that will be utilized, as well as the aggregate amount of each resource of each type (Fig. 1).

In heterogeneous environment, resource allocation is thus represented by two vectors, a maximum elementary allocation vector and an aggregate allocation vector.

Resource allocation graphs are used to detect and avoid deadlock in heterogeneous system. Creating resource allocation graph for IaaS is very difficult due to dynamic nature of cloud user and their respective requirements.

2.3 Resources Allocation Based on Approaches Fuzzy

The current, we know only a few research apply Fuzzy Logic to resource allocation. Fuzzy logic [5] is a tool that deals with uncertain, imprecise, or qualitative information, as well as with precise information in heterogeneous system. Which are not defined with formal mathematical the model ***V*** VM-out-of-***N*** PM. In Boolean logic, an element x can or cannot belong to a set A, with a membership degree equal to 1 or 0. Instead, in fuzzy logic, the membership degree of x to a fuzzy set F has a value in a continuous interval between 0 and 1. A fuzzy subset A of a set X can be defined as a set of ordered pairs, each with the first element from X and the second from the interval [0,1], with exactly one ordered pair for each element of X. Unlike classical set theory, Fuzzy theory allows an element to have partial membership degree in one or more fuzzy sets. This membership degree is obtained through membership functions that map elements into the interval [0,1]:

$$\mu A : X \rightarrow [0, 1] \tag{4}$$

The value zero and one represent complete non-membership and complete membership respectively, whereas values in the interval [0, 1] represent intermediate degrees of membership. Fuzzy logic can be used for the design of control dynamic allocation resource in heterogeneous systems. In this case, the controller is called

Fuzzy Logic Controller (FLC). The core of the FLC is the inference engine, whose role is to apply the inference rules (IF-THEN rules) contained in the rule base. IF-THEN rules are made by antecedents, consequence and embody the system control dynamic. These rules are implemented by fuzzy implications and represent the expert knowledge about the systems behavior.

The FLC main modules are the following [9]:

- Inputs fuzzification: FLC receives as input the differences between the IaaS resources values. The VM request parameters, which will be fuzzified using membership functions.
- Inference engine and Rule base: the role of the Inference engine is to infer the fuzzy implication contained in the Rule base.
- Defuzzification method: center of gravity has been used as defuzzification algorithm for the aggregated fuzzy subset.

The inputs fuzzification process includes the definition of fuzzy sets for inputs classification and the corresponding membership functions, thus determining the inputs membership degrees, always included by definition in the interval [0, 1].

3 Related Works

Resource allocation in cloud computing has attracted the attention of the research community in the last few years. In 2015, Nguyen [1] used an anlytical model to estimate completion time, and they used the result to determine right size of resources. In previous articles we have published two algorithms. Which were used to detect deadlock in resources allocation heterogeneous distributed platforms [1,3]. We provide deadlock detection algorithms and resolve the optimization problems of resources based the recovery of resources allocated. In [3], we provide deadlock detection algorithms and resolve optimal problems according to groups of users. To maximize performance, these scheduling algorithms tend to choose free load servers when allocating new VMs. On the other hand, the greedy algorithm can allocate a small lease (e.g. with one VM) to a multicore physical machine. In this study, we propose solutions to avoid deadlock in resource supply, then proceed to build the resource supply automatically detects and avoid deadlock occurs. This issue is also effective in allocation resources. The mathematical model computes the optimal number of servers and the frequencies at which they should run in [5,6,9].

In [6], the authors build a fuzzy set for only 7 cases. However, in actual, the value of fuzzy sets is more. In addition, the authors use fuzzy logic to build value for each member of the fuzzy sets. In the next study, we will propose a new model for VM resource allocation based on Hedge Algebras. We will use hedge algebras for building value for each member in the fuzzy sets and fuzzy membership function.

Based on the advantages of the structure of hedge algebra (HA), the authors studied and apply in the models (fuzzy relational database and fuzzy object-oriented database model) and there have been many results [5,6].

Approached in hedge algebra, in which linguistic semantics be quantified by quantitative semantic mapping of hedge algebra. In this approach, language semantics can be expressed in a neighborhood of intervals determined by the fuzziness measure of linguistic values of an attribute as a linguistic variable.

On this basis, we will consider domain of fuzzy value is hedge algebra and transformer interval values into subsegment [0, 1], and thereby providing resource allocation for virtual machine with fuzzy information and uncertainty become effective.

4 Our Algorithm

In this paper, we will proposed algorithm for deadlock avoidance maintains property of n-vertex directed graph when the new is added in the graph using two-way search.

The time bound for the incremental cycle algorithm for deadlock avoidance take O(m*(n − 1)/2 + 2e) time bound for the e edge insertion in the directed graph. It reports the cycle when the algorithm detects for edge (v,m) that there exist a path from vertex w to v.

As can be considered here is the case where a process p_i requires the resources it needs for its session one after the other, hence the name incremental requests. The main issue that has to be solved is the avoidance of deadlocks.

As a matter of fact that deadlock avoidance would help in providing resources as good as possible.

As it is commonly known, process must wait for the resources as deadlock occurred since they are being occupied by other processes.

Therefore, when process in a wait-for the dependency is broken, the corresponding information must be instantly restored from the system, otherwise, deadlock would be happening.

Let r_1, r_2, ...,r_n be the whole set of resources accessed by the processes, each process accessing possibly only a subset of them.

Let $<$ be a total order on this set of resources. The processes are required to obey the following rule:

- During a session, a process may invoke request resource(r_k) only if the it has already obtained all the resources r_j it needs which are such $r_j < r_k$.
- As p_1 is owning the resource r_a and waiting for the resource r_b, it invoked first request resource(r_a) and then a request resource(r_b).
- As p_2 is owing the resource x_b and waiting for the resource r_a, it invoked first request resource(r_b) and then request resource(r_a).

Algorithm 1. Deadlock Avoidance Algorithm (DAA)

Input: $P_i^{j(CPU)^*}$, $P_i^{j(RAM)^*}$ from IaaS provider i;
Output: new resource $r_j^{CPU^{(n+1)}}$, $r_j^{RAM^{(n+1)}}$;

BEGIN
Operation request resource (r_i) in the critical section is
$csstate_i \longleftarrow$ trying;
$lrd_i \longleftarrow clock_i + 1$;
for each $j \in R_i$ do
if ($usedby_i$[j] =0) the send request (lrd_i,i) to p_j end for;
$sentto_i$ [j] $\longleftarrow$ true;
$usedby_i$[j] $\longleftarrow$ R
else $sentto_i$[j] $\longleftarrow$false
end if
end for;
$usedby_i$[i] $\leftarrow k_i$;
wait ($\sum_{j=1}^{V} usedby_i[j] \leq NPM$);
csstate$_i$ $\longleftarrow$ in;
When REQUEST(i,j,k) is received from p_j do
$clock_i \leftarrow$ max($clock_i$,n);
$prio_i \leftarrow$ (csstate$_i = in$) $\vee$ (($csstate_i = trying$) $\wedge$ ((lrd_i, j, k) $<$ (n, j, k)));
if(prio$_i$) then send USED(*N* PM) to p_j else if($n_i \neq$ *N* PM) then send USED(*N* PM - n_i) to p_j end if
$permdelayed_i \leftarrow permdelayed_i \cup$ j
end if.
When permission(i,j,k) is received from p_j do
$NPM_i \leftarrow NPM_i \setminus$ j;
When USED(x) is received from p_j do $usedby_i$[j] $\leftarrow usedby_i$[j] $-x$;
if((csstate$_i = trying$) $\wedge$ ($usedby_i[j] = 0$) $\wedge$ ($notsentto_i[j]$)
then send RELEASE(lrd_i,j,k) to p_j
$sentto_i$[j] $\leftarrow$ false;
$usedby_i$[j] $\leftarrow$ *N* PM;
end if.
END.

In many systems, a heterogeneous environment is preferable to one that is homogeneous. However, it provides better performance particular systems and workload. Even if the workload itself is more suitable to a heterogeneous distributed platforms, the systems rescheduling algorithm should exploit heterogeneity well to benefit from it (Table 1).

Table 1. The description of notations

Notations	Meanings
$csstate_i$ = out	mean that p_i is not interested in executing the statemment critical section
$csstate_i$ = in	mean that p_i is executing the statemment critical section
$csstate_i$ = trying	mean that p_i is executing the operation requests resource
$clock_i$	mean that is a scalar clock initialized to 0
lrd_i	mean that is a local variable used by p_i to save the logical date of its
$prio_i$	mean that is an auxiliary Boolean when it receives a request message
$permdelayed_i$	mean that is a set used by p_i to contain the identities of the processes
$usedby_i$	mean that is the identities of the processes

5 Experiments and Results

In this paper, the solution provides effective resources is done through two algorithms. Algorithm resource requirements and algorithms avoid deadlock in resource supply. Based on the resource model provides V VM-out-of-N PM. Methods of optimizing the use of functions in the formula 3. Optimal recovery method in materials allocated because the process still holds resources when finishing requirements (Table 2).

The comparative analysis of experimental result can be seen in many times, apter task execution, although there were individual time improved PDA algorithm response time was not significantly less than an optimal time algorithm. In most cases, improved algorithm is better than the optimal time algo-

Table 2. Comparison the optimal time of our algorithm DAA to PDDA Improved [3] algorithm

Cloudlet ID	Data center ID	VM ID	PDDA Improved			DAA			Improved (%)
			Start	End	Time	Start	End	Time	
0	1	1	0.1	100.1	100	0.1	70.1	70	22.22%
1	2	2	0.1	110.1	110	0.1	80.1	80	20.00%
2	3	3	0.1	132.1	132	0.1	80.1	80	27.27%
3	3	4	0.1	145.1	145	0.1	90.1	90	43.75%
4	1	5	0.1	147.1	147	0.1	100.1	100	39.39%
5	2	6	0.1	145.1	145	0.1	110.1	110	36.05%
6	1	7	0.1	152.1	152	0.1	110.1	110	42.86%
7	2	8	0.1	153.1	153	0.1	100.1	100	44.44%
8	3	9	0.1	163.1	163	0.1	90.1	90	50.55%
9	2	10	0.1	168.1	168	0.1	65.1	65	64.86%

Table 3. Comparison the optimal time of our algorithm DAA to PDA [2] algorithm

Cloudlet ID	Data center ID	VM ID	PDA [2]			DAA			Improved (%)
			Start	End	Time	Start	End	Time	
0	1	1	0.1	60.1	60	0.1	70.1	70	10.00%
1	2	2	0.1	67.1	67	0.1	80.1	80	13.00%
2	3	3	0.1	74.1	74	0.1	80.1	80	16.00%
3	3	4	0.1	82.1	82	0.1	90.1	90	8.00%
4	1	5	0.1	83.1	83	0.1	100.1	100	23.00%
5	2	6	0.1	95.1	95	0.1	110.1	110	16.00%
6	1	7	0.1	90.1	90	0.1	110.1	110	20.00%
7	2	8	0.1	89.1	89	0.1	100.1	100	11.00%
8	3	9	0.1	72.1	72	0.1	90.1	90	18.00%
9	2	10	0.1	50.1	50	0.1	65.1	65	15.00%

rithm, thus validated the correctness and effectiveness. The process of rescheduling parallel tasks determines the order of task execution and the processor to which each task is assigned.

Typically, an optimal reschedule is achieved by minimizing the completion time of the message request.

Finding the optimal reschedule has long been known as an NP-hard problem in both heterogeneous distributed platforms and homogeneous.

The experiments were conducted in an environment CloudSim, test data are as follows: Table 3 [11].

The experiment results show that our technology resource allocation used algorithms avoid deadlock can reduce completion time by up 20% when compared to simple algorithm allocation (Fig. 2).

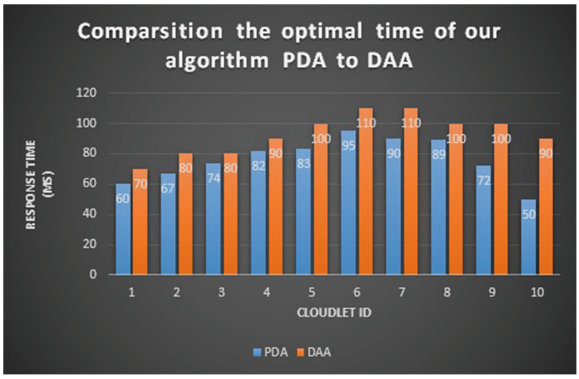

Fig. 2. Comparison the optimal time of algorithms PDA and DAA

6 Conclusion and Future Works

A wait-for graph in model V VM-out-of-N PM is a directed graph used for detection and avoidance deadlock. The comparative analysis of experimental result can be seen in many times, apter task execution, although there were individual time DDA algorithm response time greater than an optimal time algorithm, in most cases, improved algorithm is better than the optimal time algorithm, thus validated the correctness and effectiveness. In summary, there have been a body of researches in cloud computing systems that take either heterogeneous.

In paper, we attempt to exploit heterogeneity and ensure fairness at the same time. A deadlock avoidance algorithm is implemented for resource allocation on heterogeneous distributed platforms. The avoid deadlock algorithm has $O(m^*(n-1)/2 + 2e)$ time complexity, an improvement of approximate orders of magnitude in practical cases.

In this way, programmers can quickly avoid deadlock and then resolve the situation, e.g., by releasing held resources. Through this research, we found that the application of appropriate avoid deadlock algorithms would give optimal performance to distributed resources of virtual server systems.

References

1. Nguyen, H.H.C., et al.: A new technical solution for resource allocation in heterogeneous distributed platforms. In: Mizera-Pietraszko, S.F.J. (ed.) Advances in Digital Technologies, pp. 184–194. IOS Press, The Netherlands: The University of Macau, Macau (2015)
2. Nguyen, H.H.C., et al.: Deadlock prevention for resource allocation in heterogeneous distributed platforms. In: Mizera-Pietraszko, S.F.J., Chung, Y.-L., Pichappan, P. (eds.) Advances in Digital Technologies, pp. 40–49. IOS Press, The Netherlands: The University of Macau, Macau (2016)
3. Nguyen, H.H.C., Dang, H.V., Pham, N.M.N., Le, V.S., Nguyen, T.T.: Deadlock detection for resource allocation in heterogeneous distributed platforms. In: Unger, H., Meesad, P., Boonkrong, S. (eds.) Recent Advances in Information and Communication Technology 2015. AISC, vol. 361, pp. 285–295. Springer, Cham (2015). doi:10.1007/978-3-319-19024-2_29
4. Nguyen, H.H.C., Le, V.S.: Detection and avoidance deadlock for resources allocation in heterogenenous distributed plaforms. Int. J. Comput. Sci. Telecommun. (IJCST), English **6**(2), 1–6 (2015)
5. Adami, D., Gabbrielli, A., Giordano, S., Pagano, M., Portaluri, G.: A fuzzy logic approach for resources allocation in cloud data center. In: Proceedings 2015 IEEE Globecom Workshops (GC Wkshps), pp. 1–6 (2015)
6. Thang, D., Quoc, D.C.: Defining membership functions in fuzzy object-oriented database model. In: Dang, T.K., Wagner, R., Küng, J., Thoai, N., Takizawa, M., Neuhold, E. (eds.) FDSE 2015. LNCS, vol. 9446, pp. 314–322. Springer, Cham (2015). doi:10.1007/978-3-319-26135-5_23
7. Sotomayor, B.: Provisioning computational resources using virtual machines and leases. Ph.D. thesis, University of Chicago (2010)

8. Sotomayor, B., Keahey, K., Foster, I.T.: Combining batch execution and leasing using virtual machines. In: HPDC, pp. 87–96 (2008)
9. Warneke, D., et al.: Exploiting dynamic resource allocation for efficient parallel data processing in the cloud. IEEE Trans. Parallel Distrib. Syst. **22**(6), 985–997 (2011)
10. Kshemkalyani, A.D., Singhal, M.: Distributed Computing Principles, Algorithms, and Systems. Cambridge University Press, UK (2008)
11. www.cloudbus.org/cloudsim/