International Journal of Innovative Technology and Interdisciplinary Sciences
www.IJITIS.org
ISSN: 2613-7305
Volume 2, Issue 1, pp. 98-107, 2019
DOI: https://doi.org/10.15157/IJITIS.2019.2.1.98-107
Received October 12, 2018; Accepted November 16, 2018

Avoid Deadlock Resource Allocation (ADRA) Model V VM-out-of-N PM

Ha Huy Cuong Nguyen [*a1], **Van Thang Doan** [b2]

[1]Department of Information Technology, Quang Nam University, Vietnam

[2]Industrial University of Ho Chi Minh, Ho Chi Minh City, Vietnam
*[a] nguyenhahuycuong@gmail.com; [b] vanthangdn@gmail.com

ABSTRACT

This paper presents an avoid deadlock resource allocation (ADRA) for model V VM-out-of-N PM since cloud computing is a new computing paradigm composed of grid computing, distributed computing and utility concepts. Cloud computing presents a different resource allocation paradigm than either grids or distributed systems. Cloud service providers dynamically scale virtualized computing resources as a service over the internet. Due to variable number of users and limited resources, cloud is prone to deadlock at very large scale. Resource allocation and the associated deadlock avoidance is problem originated in the design and the implementation of the distributed computing, grid computing. In this paper, a new concept of free space cloud is proposed to avoid deadlock by collecting available free resource from all allocated users. New algorithms are developed for allocating multiple resources to competing services running in virtual machines on a heterogeneous distributed platform. An experiment is tested in CloudSim. The performance of resource pool manager is evaluated by using CloudSim and resource utilization and indicating good results.

Keywords: Cloud computing, IaaS, Resource allocation, Heterogeneous, Distributed platforms, Avoid deadlock.

1. INTRODUCTION

In the past, grid computing and batch scheduling had both been commonly used for large scale computation [1]. Cloud computing presents a different resource allocation paradigm than either grids or batch schedulers [2],[3],[5],[6]. Cloud services are provided by cloud service provider and are used by cloud customer on a pay per use basis. Therefore, QoS as well as reliability are important aspects from user. Infrastructure as a Service is the concept of providing Hardware as a Service. It provides the required hardware as CPU, memory (RAM), storage (HDD). IaaS is the Virtual Machine (VM) in the heterogeneous environment. There are number of successful IaaS providers such as Amazon, GoGrid and FlexiScale. While a number of recent papers address virtualization of enterprise applications, such as resource virtualization [4],[5], on-demand resource provisioning management based on virtual machines [6],[7], and QoS management of virtual machine [8]. These works lead to improvements in the performance of virtualization and resource utilizations. Since IT infrastructure customer requirements for cloud infrastructure services are varied, infrastructure providers have to ensure that they can be flexible in their service delivery while keeping the infrastructure service customers efficiently increasing ability of commodity hardware to run applications within VMs. VMs allow both the isolation of

applications from the underlying hardware and other VMs, and the customization of the infrastructure resource to meet the requirements of the IT infrastructure customer. Virtualized datacenters are exposed to many challenges such as quality of service, efficient resource utilization, deadlocks, task scheduling and dynamic resource allocation [5]. Though numbers of cloud users are increasing abruptly in last years but cloud is still maturing in total number of resource and effective allocation strategies. Deadlock in resource allocation heterogeneous distributed platforms due to over demand of resources [3],[4]. In this paper, a method to avoid a deadlock occurs in the process of providing resources in class infrastructure as a service IaaS has been developed. 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. Updated researches on the issue are referred to in [18], [19], [20], and [21].

The paper is organized as follows: section 2 provides the related works; section 3 describes existing models; section 4 presents approaches and section 5 presents our conclusions and suggestions for future work.

2. SYSTEM MODEL RESORCE ALLOCATION IN HETEROGENOUS DISTRIBUTED PLATFORMS

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

Resource allocation involves deciding what, how many, where, and when to make the resource available to the user. Typically users decide the type and amount of the resource containers to request, then providers place the requested resource containers onto nodes in their Data Centers. In a heterogeneous environment, some tasks run faster on a particular node than others. To take a full advantage of available hardware, cloud oriented applications must be heterogeneous-aware.

A heterogeneous distributed platforms are composed of a set of an asynchronous processes $\{p_1, p_2, \ldots, p_n\}$ that communicates by message passing over the communication network [6]. Authors in [14] have the same opinions that the requested resource model of distributed system is divided into five model resource requirements. It is simple resource model: 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 V VM-out-of-N PM problem depicts on-demand resource allocation to V VM residing in 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_j at time t, where

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

E_{ijt} obeys the rules as follows:

$$E \geq \sum_{i=1}^{N} \sum_{j=1}^{V_N} E_{ijt} \quad (2)$$

$$E_{ijt} \geq C_{ij} \geq 0 (i = 1,...,N; j = 1,...,V_N)$$

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:

$$F_t = \min \sum_{i=1}^{N} \sum_{i=1}^{V_i} \frac{f_{ij}(EN_{ijt}, \sum_{x=1}^{V} EO_{ijt}^{x}, D_{ijt})}{} \times SP_{ij}$$

$$\begin{cases} \sum_{i=1}^{N} \sum_{i=1}^{V_i} E_{ijt} \le E \\ E_{ijt} \ge C_{ij} (i = 1, 2, ..., V; j = 1, 2, ..., N) \\ \sum_{i=1}^{V_i} EN_{ijt} + \sum_{j=1}^{V_i} EO_{ijt}^{i} \le E_i \\ E_{it} \ge C_{ij} (i = 1, 2, ...V; j = 1, 2, ..., N) \end{cases} \quad (3)$$

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

2.2 Wait for Graph (WFG)

In heterogeneous distributed platforms systems, the state of the system can be modeled by directed graph, called a wait for graph (WFG) [14]. In a WFG, nodes are processors and there is a directed edge from node P_1 to mode P_2 if P_1 is blocked and is waiting for P_2 to release some resource. A system is deadlocked if and only if there exists a directed cycle or knot in the WFG. In making a resource allocation decision, performance prediction plays an important role. In paper, we use performance prediction to estimate the completion time of a lease with given resources, and to determine whether the lease will be finished by the deadline with the given amount of resources (Figure 1).

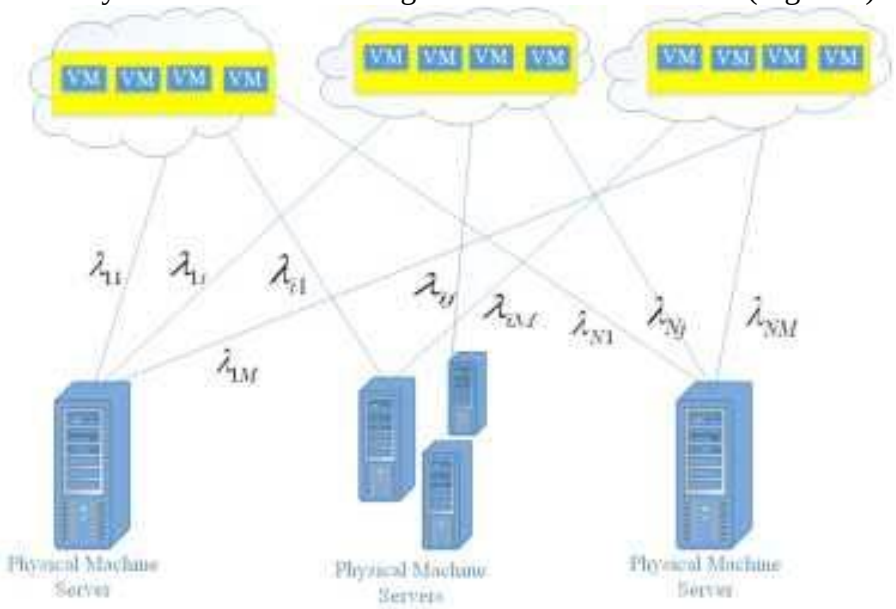

Figure 1. A simple platforms

A set P = {P_1, P_2 ,P_k} of k > 1 entities is deadlocked when the following two conditions simultaneously hold:

- Each entity P_i. P is waiting for an event permission that must be generated from another entity in the set;

- No entity P_i. P can generate permission while it is waiting.

If these two conditions hold, the entities in the set will be waiting forever, regardless of the nature of the permission and of why they are waiting for the permission; for example, it could be because P_i needs a resource held by P_j in order to complete its computation. A useful way to understand the situations in which deadlock may occur is to describe the status of the entities during a computation, with respect to their waiting for some events, by means of a directed graph, called wait-for graph [14].

2. 3 Problem identification

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 [14].
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 [14]. 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. An allocation is thus represented by two vectors, a maximum elementary allocation vector and an aggregate allocation vector.

3. RELATED WORKS

Resource allocation in cloud computing has attracted the attention of the research community in the last few years. A vast amount of research has been conducted on resource allocation, especially in the high performance computing community. Many other works are similar, basically matching the specification to resources. However, user resource requirements do guarantee the optimal resource allocation, as the might be no available resources match to the specification. Nguyen et al [3] used an analytical model to estimate completion time, and they used the result to determine right size of resources. In [8] [9], Armbrust, Srikantaiah and et al studied the problem of request scheduling for multi-tiered web applications in virtualized heterogeneous systems in order to minimize energy consumption while meeting performance requirements. They proposed a heuristic for a multidimensional packing problem as an algorithm for workload consolidation. In previous articles we have published two algorithms. Which were used to detect deadlock in resources allocation heterogeneous distributed platforms [3] [5]. We provide deadlock detection algorithms and resolve the optimization problems of resources based the recovery of resources allocated. In [5], we provide deadlock detection algorithms and resolve optimal problems according to groups of users. Most of the studies were set to study scheduling policy effectiveness in resources allocation. Sotomayor et al. [10] [11] proposed a lease-based model and implemented First-Come-First-Serve (FCFS) scheduling algorithm and a greedy based VM mapping algorithm to map leases that include some of VMs with/without start time and user specified duration to a set of homogeneous physical machines (PMs). Much of the research conducted homogeneous physical machines (PMs). Not much research provide resources for heterogeneous systems. 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 multi-core physical machine. In this study, we propose solutions to detect 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 [12] and a new approach for dynamic autonomous resource management in computing clouds introduced in [13].

4. PROPOSED ALGORITHM

In this paper, we will approach a 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 exists a path from vertex w to v.

As can be considered here is the case where a process pi 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. For instance, the processes P_8 and P_2 in Fig.1 can be taken in consideration when both processes want to acquire both the resources r_1 and r_2. 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. However, achieving effectively resource scheduling would greatly lessen such worse situation of deadlock.

Therefore, proposed algorithm works as engine springing up the needed information about the process situations whether each one is in underway, lack or waiting and this fact could be graphically expressed through dependence graph presented in this paper. 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= $\{r_1, r_2, \ldots, r_n\}$ be the whole set of resources accessed by the processes, each process accessing possibly only a subset of them. Let R < 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 f_j it needs which are such $f_j < r_k$.
- As p_1 is owning the resource ra and waiting for the resource rb, it invoked first request resource (r_a) and then a request resource(r_b).
- As p_2 is owing the resource xb and waiting for the resource ra, it invoked first request resource (r_b) and then request resource(r_a).

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 platform, the systems rescheduling algorithm should exploit heterogeneity well to benefit from it (Table 1 and Table 2).

Table 1. Requests resources for the model V VM-out-of-N PM

Input: P_i request resources from IaaS provider VM_i
Output: new resource grant VM_i
Begin
Operation request resource (r_i) in the critical section is
$csstate_i \leftarrow trying$;
$lrd_i \leftarrow clock_i + 1$;
For each $j \in R_i$ do
If ($usedly_i[j]=0$) the send request (lrd_i,i) to p_j end for;
$Sentto_i[j] \leftarrow true$;
$usedby_i[j] \leftarrow R$
else $sentto_i[j] \leftarrow false$
end if
end for;
$usedby_i[j] \leftarrow k_i$;
wait($(\sum_{i=1}^{V} usedby_i[j] \leq NPM)$;
$csstate_i \leftarrow in$;
Operation release resource (r_i) in the critical section is
$csstate_i \leftarrow out$;
for each $j \in permdelayedi$ do send permission(i,j) to pj end for;
$Ri \leftarrow permdelayedi$;
$Permdelayedi \leftarrow \varnothing$
End.

Table 2. Avoid deadlock resource allocation for the model V VM-out-of-N PM

Input: P_i request resources from IaaS provider VM_i
Output: new resource grant VM_i
Begin
When REQUEST(i,j,k) is received from p_j do
$clock_i \leftarrow max(clock_i,n)$;
$prio_i \leftarrow (csstate_i=in) \vee ((csstate_i=trying) \wedge ((lrd_i,i)<(n,j)))$;
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.
When permission(i,j,k) is received from p_j do
N $PM_i \leftarrow N\ PM_i \setminus j$;
When USED(x) is received from p_j do $usedby_i[j] \leftarrow -x$;
If (($csstate_i=trying$) $\wedge$ (($usedly_i[j]=trying$) $\wedge$ ($notsentto_i[j]$)
Then send RELEASE(i,jk) to p_j
$sentto_i[j] \leftarrow true$;
$usedby_i[j] \leftarrow N\ PM$;
end if
END.

5. EXPERIMENTS AND RESULTS

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. The experiments were conducted in an environment CloudSim, test data are as follows: The experiment results show that our technology resource allocation used algorithms avoid deadlock can reduce completion time by up 30 percent when compared to simple algorithm allocation (Table 3).

The comparative analysis of experimental result can be seen in many times, after 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 algorithm, 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 (Figure 2).

Table 3. Comparison the optimal time of our algorithm ADRA to PDDA

			Algorithm PDDA improved			and ADRA			
Cloudlet ID	Datacenter ID	VM ID	Start	End	Time	Start	End	End	Time Improved (%)
0	1	1	0.1	100.1	100	0.1	60.1	60	18.22%
1	2	2	0.1	110.1	110	0.1	70.1	70	17.00%
2	3	3	0.1	132.1	132	0.1	70.1	70	26.00%
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	35.05%
5	2	6	0.1	145.1	145	0.1	104.1	104	40.00%
6	1	7	0.1	152.1	152	0.1	104.1	104	42.86%
7	2	8	0.1	153.1	153	0.1	110.1	110	45.65%
8	2	9	0.1	163.1	163	0.1	105.1	105	50%
9	2	10	0.1	168.1	168	0.1	111.1	11	55%
10	3	11	0.1	170.1	170	0.1	110.1	110	56%
11	2	12	0.1	169.1	169	0.1	115.1	115	57%
12	1	13	0.1	172.1	172	0.1	120.1	120	58%
13	1	14	0.1	162.1	162	0.1	100.1	100	38%
13	2	15	0.1	152.1	152	0.1	100.1	100	38%

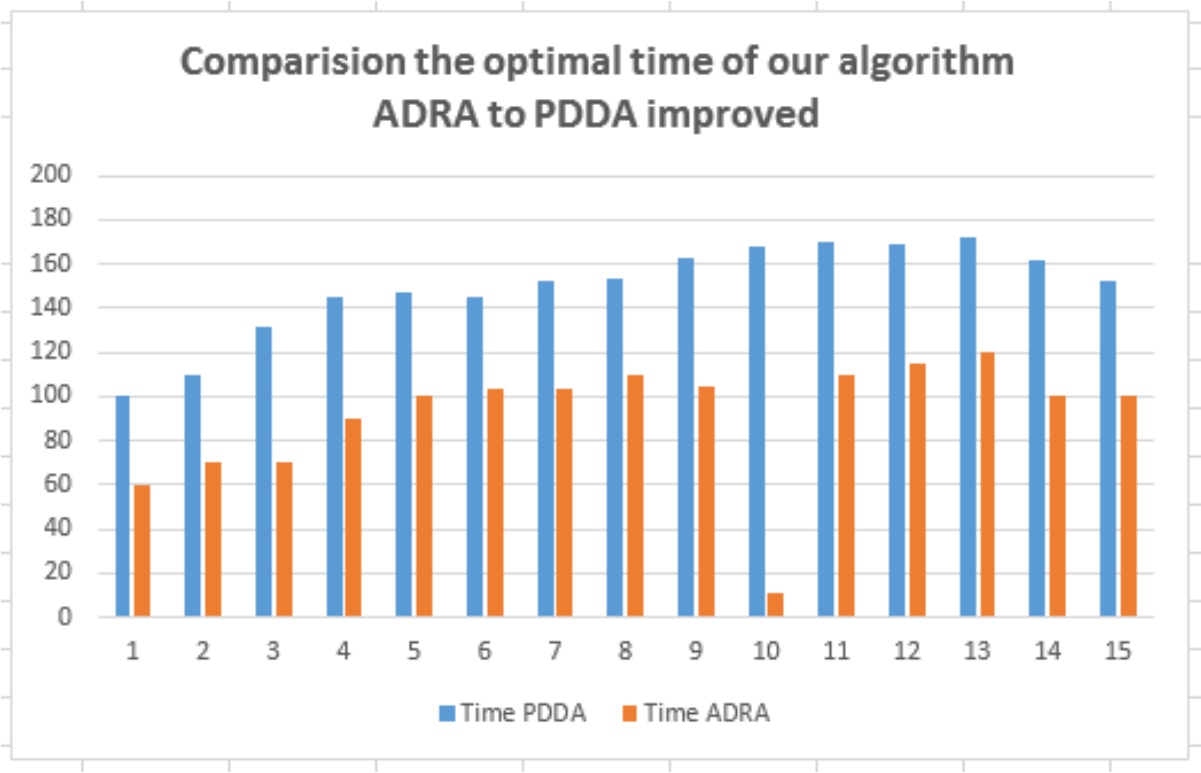

Figure 2. Comparison of the ADRA vs PDDA.

6. CONCLUSIONS

The comparative analysis of experimental result can be seen in many times, after task execution, although there were individual time ADRA algorithm response time greater than an optimal time algorithm, in most cases, improved algorithm is better than the optimal time algorithm, thus validated the correct-ness 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. Our main approach focuses on applying deadlock avoidance algorithms for each type of lease contracts and applying the proposed algorithm in resource allocation on heterogeneous distributed platform. Through this research, we found that the application of appropriate avoid deadlock algorithms would give optimal performance to distributed resources of virtual server systems.

ACKNOWLEDGMENT

The authors would like to thank Quang Nam University and Industrial University of Ho Chi Minh for supporting this research project code: 182.CNTT02/HD-DHCN.

CONFLICT OF INTERESTS

The authors declare that there is no conflict of interest regarding the publication of this research article.

REFERENCES

[1] Vouk, M.A., Cloud computing - issues, research and implementations, In: Information Technology Interfaces. ITI 2008, Dubrovnik, Croatia, 2008, 31-40

[2] Yazir, Y.O., Matthews, C., Farahbod, R., Dynamic resource allocation in computing clouds using distributed multiple criteria decision analysis, In: IEEE 3rd International Conference on Cloud Computing, Florida, USA, 2010, 91-98

[3] Bo Z., Shuang X., Yang A., Wei T., A dynamic priority task scheduling strategy for TSS deadlock based on value evaluation, China Communications, 2016, 13(1), 161-175

[4] Wang Y., Hu M., Kent K.B., An effective admission control scheme with deadlock resolutions for workflow scheduling in clouds, Computing, 2015, 97(4), 379-402

[5] Lim J., Suh T., Yu H., Unstructured deadlock detection technique with scalability and complexity efficiency in clouds, International Journal of Communication Systems, 2014, 27(6), 852-870

[6] Ha Huy Cuong Nguyen, et al., A New Technical Solution for Resource Allocation in Heterogeneous Distributed Platforms, in Advances in Digital Technologies, S.F. Jolanta Mizera-Pietraszko, Editor. 2015, IOS Press,The Netherlands: The University of Macau, Macau, 2015, 184 - 194

[7] Ha Huy Cuong Nguyen, et al., Deadlock Prevention for Resource Allocation in Heterogeneous Distributed Platforms, in Advances in Digital Technologies, S.F. Jolanta Mizera-Pietraszko, Yao-Liang Chung, Pit Pichappan, Editor. 2016, IOS Press,The Netherlands: The University of Macau, Macau, 2016, 40-49

[8] Ha Huy Cuong Nguyen, et al., Deadlock Detection for Resource Allocation in Heterogeneous Distributed Platforms, in Recent Advances in Information and Communication Technology 2015, H. Unger, Editor. Springer International Publishing, Switzerland, 2015, 285 - 295

[9] Ha Huy Cuong Nguyen, Van Son Le. Detection and avoidance deadlock for resources allocation in heterogenenous distributed plaforms, International Journal of Computer Science and Telecommunications(IJCST), 2015, 6(2), 1-6

[10] Amazon EC2, https://aws.amazon.com/ec2/.

[11] Armbrust, M., Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., Patterson, D.,Rabkin, A., Stoica, I., Zaharia, M., A view of cloud computing, Commune ACM, 2010, 53(4), 50-58

[12] Srikantaiah, S., Kansal, A., Zhao, F., Energy aware consolidation for cloud computing. Cluster Computer, 2009, 12, 1-15

[13] Sotomayor, B.: Provisioning Computational Resources Using Virtual Machines and Leases. Ph.D. thesis, University of Chicago, 2010

[14] Sotomayor, B., Keahey, K., Foster, I.T., Combining batch execution and leasing using virtual machines. In: HPDC, New York, USA, 2008, 87-96

[15] Warneke et al, Exploiting dynamic resource allocation for efficient parallel data processing in the cloud. IEEE Trans. Parallel Distributed System, 2011, 22(6), 985-997

[16] Ajay D. Kshemkalyani, M.S., Distributed Computing Principles, Algorithms, and Systems. 2008, UK: Cambridge University Press.

[17] www.cloudbus.org/cloudsim/

[18] Vu Trieu Minh, Riva Khanna, "Application of Artificial Intelligence in Smart Kitchen", *International Journal of Innovative Technology and Interdisciplinary Sciences*, Vol. 1, no. 1, pp. 1-8, Nov. 2018.

[19] Vu Trieu Minh, Reza Moezzi, Minh, Mart Tamre, "Fuzzy Logic Control for a Ball and Beam System", *International Journal of Innovative Technology and Interdisciplinary Sciences*, Vol. 1, no. 1, pp. 39-48, Nov. 2018.

[20] Ivan Ovchinnikov, Pavel Kovalenko, "Predictive Control Model to Simulate Humanoid Gait", *International Journal of Innovative Technology and Interdisciplinary Science*s, Vol. 1, no. 1, pp. 9-17, Nov. 2018.

[21] Mart Tamre, Robert Hudjakov, Dmitry Shvarts, Ahti Polder, Maido Hiiemaa, Mart Juurma, "Implementation of Integrated Wireless Network and MATLAB System to Control Autonomous Mobile Robot", *International Journal of Innovative Technology and Interdisciplinary Sciences*, Vol. 1, no. 1, pp. 18-25, Nov. 2018.