ISSN 1598-0170 (Print)
ISSN 2287-1136 (Online)
http://www.jksii.or.kr

퍼지 분류 및 동적 임계 값을 사용한 적응형 VM 할당 및 마이그레이션 방식☆

Adaptive VM Allocation and Migration Approach using Fuzzy Classification and Dynamic Threshold

존크리스토퍼 마테오[1] 이 재 완[2*]
John Cristopher A. Mateo Jaewan Lee

요 약

클라우드 컴퓨팅이 발전하면서, 전체적인 관리 비용을 최소화하기 위해 자원 관리 기술이 중요하다. 클라우드 환경에서 사용자 선호도에 기반한 호스트의 활용과 가상머신들의 요구사항은 본질적으로 자주 바뀐다. 이러한 문제를 해결하기 위해, 호스트와 가상머신들이 분류가 되지 않은 상황에서 효율적인 자원 할당 방법을 연구할 필요가 있다. 에너지 소비를 절약하기 위해 액티브 호스트를 줄일 때, 가상머신들을 다른 호스트로 이주할때 임계값을 사용한다. 가상머신의 자원 요구량과 호스트의 자원 이용량을 분류할 때 Fuzzy Logic을 이용하여 적응성 가상머신 할당 및 이주 방법을 제안한다. 제안한 방법은 자원의 요구량에 따라 가상머신들을 분류한 뒤 가장 적은 자원활용도를 갖는 호스트에게 자원을 할당하며, 과부하된 호스트들로부터 가상머신을 이주시킬 때 상위 임계치를 설정하기 위해 각 호스트들의 자원 활용도가 사용된다. 이주하기 위한 후보 가상머신들을 선택할 때, 호스트에서 높은 자원을 가진 가상머신을 선택한다. 시뮬레이션을 통해 연구 결과를 평가하였고, 평가 결과 다른 가상머신 할당 방법들보다 효율적임을 증명하였다.

☞ 주제어 : 클라우드 컴퓨팅, 클라우드 컴퓨팅, 가상 컴퓨터 할당 및 마이그레이션, 퍼지 논리

ABSTRACT

With the growth of Cloud computing, it is important to consider resource management techniques to minimize the overall costs of management. In cloud environments, each host's utilization and virtual machine's request based on user preferences are dynamic in nature. To solve this problem, efficient allocation method of virtual machines to hosts where the classification of virtual machines and hosts is undetermined should be studied. In reducing the number of active hosts to reduce energy consumption, thresholds can be implemented to migrate VMs to other hosts. By using Fuzzy logic in classifying resource requests of virtual machines and resource utilization of hosts, we proposed an adaptive VM allocation and migration approach. The allocation strategy classifies the VMs according to their resource request, then assigns it to the host with the lowest resource utilization. In migrating VMs from overutilized hosts, the resource utilization of each host was used to create an upper threshold. In selecting candidate VMs for migration, virtual machines that contributed to the high resource utilization in the host were chosen to be migrated. We evaluated our work through simulations and results show that our approach was significantly better compared to other VM allocation and Migration strategies.

☞ keyword : Cloud Computing, Virtual Machine Allocation and Migration, Fuzzy Logic

1. Introduction

Cloud Computing provides millions of users' access to cloud services on a pay-as-you-go basis, where the amount and type of resources can be modified according to the users' needs. Varieties of services are available, such as Software-as-a-Service, where the users are given access to applications that are executed on clouds, and Infrastructure-as-a-Service, which allows users to adjust their execution environment, including the hardware resources, operating system, etc. The resources available in cloud computing [1] are scalable because endless pools of servers can be added to accommodate millions of users. This means that data centers often consume large amounts of energy at peak times because a lot of hosts have to be turned on to accommodate the workload.

1 Department of Information and Communication Engineering, Kunsan National University, Korea
* Corresponding author (jwlee@kunsan.ac.kr)
[Received 17 March 2017, Reviewed 26 May 2017, Accepted 20 June 2017]
☆ This paper was supported by the research funds of Kunsan National University
☆ 본 논문은 2016년도 한국인터넷정보학회 추계학술발표대회 우수 논문 추천에 따라 확장 및 수정된 논문임.

Resource managers in cloud systems need crucial information on the hosts and virtual machines such as information used to manage cloud resources. Different resources are available in the cloud [2]. This selection includes a) Cloud Selection, b) Datacenter Selection, and c) Server Selection. In Server selection, the resources are hosts and virtual machines. Tasks are executed in virtual machines, to occupy and utilize the host's resources. Each virtual machine requires hardware resource from the host, which are CPU, memory, bandwidth, storage, etc. These can be viewed as a bin packing optimization, where bins are hosts occupied by virtual machines. The goal of the optimization is to pack the items in bins such that a minimum number of bins used are used.

This paper proposes an efficient way of a) allocating virtual machines to hosts, b) creating a dynamic upper threshold for over utilized host detection, and c) selecting virtual machines from over-utilized hosts to maintain balance in resource utilization. Fuzzy logic was used to determine weights of the virtual machine's request on resource and the host's resource utilization. Based on the highest weight on the virtual machine, it is assigned to host which has the lowest resource utilization. An upper threshold is created using the host's weights, and selecting virtual machines which utilize the most resource parameter in the host and proceeds to select candidates until the all the resource utilization weight is less than the threshold.

2. Related Works

2.1 Virtual Machine Allocation

Virtual machine allocation strategies in cloud computing are implemented to minimize the overall active hosts in cloud environments, therefore reducing the energy consumption of the data center. Each virtual machine should be evaluated and matched to a host that will allocate the VMs request for resources. These are some research that evaluates the resource utilization of the host in creating a VM allocation approach. Gupta et al. [3] proposed a mechanism for a proper placement of virtual machines to hosts, which include balancing the resource utilization of the host to the VMs. This can be achieved by classifying the physical hosts according to their resource utilization using clustering. The CPU, Memory, and Bandwidth utilization are used to cluster the hosts. Each cluster would determine which resource is available the most. Results from the simulation showed that the approach is efficient in balancing each host's resource utilization. This inspired the research to come up with another way of creating a new approach for balancing the host's resource utilization. Another approach for classifying was proposed by Shelar et al. [4], which suggested categorizing each host to different types. Virtual machines are allocated to the hosts that are considered as Target or hosts with available resources for allocation. Once the host's resource utilization reaches a certain level, it is classified as another type, the Contented or stable host. However, in this case, only the CPU and memory is included in evaluating each host. To deal with the process of allocating VMs to the hosts, Beloglazov et al. [5] suggested a placement algorithm based on the Modified Best Fid Decreasing allocation. This algorithm assigns each VM to the host by estimating the energy consumption when the VM is allocated to the host. In allocating VMs, only the CPU utilization is included.

This paper proposed a VM allocation approach based on virtual machine's resource request and available host's resource utilization which is the result of the Fuzzy Logic established here. Our approach matches virtual machines with the highest weight on its resource request to the available host with the least resource utilization weight that matches the virtual machines highest request weight.

2.2 Virtual Machine Migration

For the VM migration mechanisms, each host is evaluated first, to determine whether they are overloaded or under loaded in terms of resource utilization. Once the host is considered to be overloaded, selecting the VMs from that host will help to lower the utilization, making it stable. Using upper thresholds to determine if the resource utilization in hosts are over utilized are considered to be efficient in looking for overloaded hosts and selecting VMs for migration. Alboaneen et al. [6] proposed an overloaded detection policy with the use of upper CPU utilization threshold, which is based on the mean CPU utilization of the VMs in the host and a safety parameter constant. Detecting overloaded hosts was done by checking the host's current CPU utilization to the upper threshold given. In selecting the virtual machines in overloaded hosts, a policy called Maximum Requested Bandwidth, which chooses VMs according to the

requested bandwidth until the host is stable. In choosing for the VMs, only the requested bandwidth is included. Therefore, our approach utilizes all of the 3 important resources in both the host and VM, which is CPU, Memory, and Bandwidth. Beloglazov et al. [7] explained that utilizing fixed upper threshold is not suitable for dynamic environments. They implemented a threshold based on the host's CPU's recent utilization for detecting overloaded hosts. For selecting VMs for migration, each VM was evaluated to the upper threshold and current utilization. The goal of the approach is to select hosts that will offer the most available resources. Elijorde et al. [8] suggested on using dynamic upper and lower thresholds based on the host's recent utilization history. Once the upper threshold is established to look for overloaded hosts, the VM selection for migration includes the VM's recent resource utilization history. This is included to prevent VMs from being migrated while it is currently utilizing a huge amount of resources and to prevent future VMs from disrupting processes. Other parameters for the resource utilization should be included when utilizing the threshold.

A dynamic threshold for determining overloaded hosts, using the resource utilization weights obtained from the Fuzzy logic controller. For selecting the virtual machines from an overloaded host, an algorithm is proposed that selects each virtual machine from the host until the weights of resource utilization of the host are below the upper threshold.

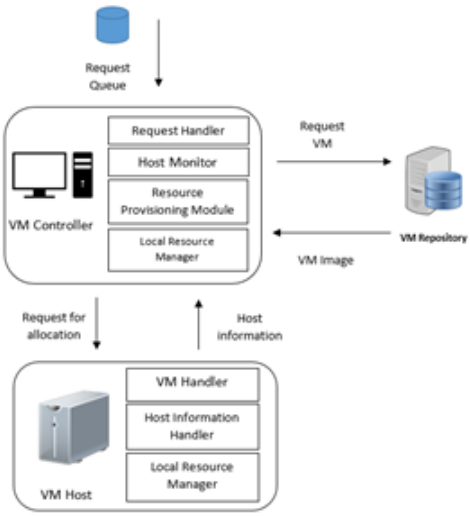
3. System Model and Methods

3.1 System Architecture

The cloud resource management system consists of a queue of tasks waiting to be assigned to a virtual machine. A host controller that handles the hosts which accept virtual machines, selects which of this host are going to be used, and assigns tasks to specific virtual machines. Virtual machine images are stored in the VM repository. The host updates the controller their resource information, current utilization. The architecture is described in Figure 1. Each module in the figure is described as follows:

A) VM Controller

Request Handler – this module assigns the request from the queue to a virtual machine that will satisfy the deadline of the tasks. It will find the specific virtual machine from the VM repository, then retrieves the VM image.



(Figure 1) System architecture of the VM Management System

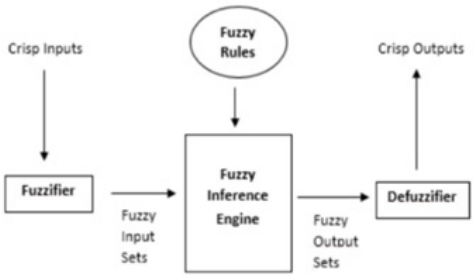
Host Monitor – this module retrieves the information from hosts regarding their resource utilization, the number of virtual machines allocated, available resources, etc. Hosts that are overloaded and under loaded are determined by this module. This information is used by the Resource Provisioning Handler in deciding which available hosts the virtual machine is allocated.

Resource Provisioning Handler – this module uses the information from the Host monitor and the virtual machine's request on resources to find the host that will allocate the host. It will find a matching host for the virtual machine that will satisfy the SLA.

B) VM Host

VM Handler – this module receives the virtual machine request from the VM Controller once it is assigned to a virtual machine. It will send a confirmation back to the controller to confirm or reject the request.

Host Information Handler – this module monitors the current status of the host. The current utilization, the number of virtual machines currently allocated, available resources for the



(Figure 2) Fuzzy Logic System

virtual machine to use are monitored. This information is sent to the controller to be analyzed for the decision-making process.

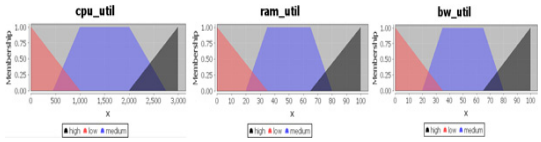
3.2 Adaptive VM Allocation/Migration on Hosts using Fuzzy Logic

Fuzzy Logic [9] is a reasoning method of computers that resembles human reasoning. The Fuzzy Logic imitates the way of decision making in humans that involve all the possible outcomes/values between YES-NO, 0-1. A Fuzzy Logic system consists of 4 parts as shown in Figure 2, namely the Fuzzifier, Fuzzy Inference Engine, Fuzzy Rules, and Defuzzifier. The Crisp Inputs are numeric values, which comes from the parameters of virtual machines and hosts in this paper. The Fuzzifier converts the Crisp Inputs into Fuzzy Sets, which are non-numeric values or linguistic terms that are associated with the crisp input.

A membership function is used to convert the crisp inputs to fuzzy sets, and vice versa. This function is used in the Fuzzification and Defuzzification process. A membership function is used to quantify the linguistic terms. In this paper, we chose the Trapezoidal membership function to quantify the linguistic terms.

Once the fuzzy input sets have been created from the crisp inputs using the membership function, it is fed to the Fuzzy Inference Engine, or sometimes called Reasoning Engine, where it is subjected to a set of Fuzzy Rules to make an output based on the rules. A Fuzzy Rule is a simple IF-THEN with a condition and a conclusion.

When a conclusion is obtained from the Fuzzy Interference Engine, a fuzzy output set is obtained by the use of Defuzzification and the output is converted to a crisp value.



(Figure 3) Membership functions for the linguistic partition of resource utilization in hosts.

Defuzzification uses the output variable according to the membership function. To obtain the crisp output, Center of Gravity (COG) is used, which is similar to calculating the center of gravity for physics. The weighted average of the membership function or the center of gravity of the area is computed as the crispest value.

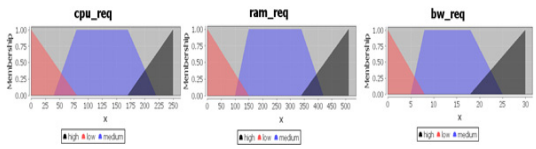
3.2.1 Classification of Host's Resource Utilization Weight using Fuzzy Logic

To classify active hosts in data centers, fuzzy logic is used to determine the weights of resource utilization. Parameters of the host include the available CPU, available RAM, and available Bandwidth, denoted by $h_i=\{CPU_U,RAM_U,BWD_U\}$. The input variables are a) utilized CPU (CPU_U), b) utilized memory (RAM_U), and c) utilized bandwidth (BWD_U).

The output variables in this controller are the a) weight of CPU utilization (CPU_W) b) weight of memory utilization (RAM_W) and c) the weight of bandwidth utilization (BWD_W). The membership functions used in classifying the hosts are shown in Figure 3 and are comprised of linguistic terms LOW, MEDIUM, and HIGH. 27 rules with different states of inputs and outputs were defined in the Fuzzy Reasoning Engine. Once a rule is chosen that matches the Fuzzy sets, the next step is to defuzzify the output of said rule to obtain the weights. These weights will determine the level of utilization on each host. These are also used to allocate of virtual machines.

3.2.2 Adaptive VM Allocation

VM Allocation is assumed to be a bin packing problem, consisting of items that must be placed in a set of bins, with an objective of minimizing the number of bins. We adopted a mechanism that enables virtual machines to be allocated properly in hosts with enough resources to accommodate the



(Figure 4) Membership functions for the linguistic partition of virtual machine resource request.

virtual machine. As the user's request varies in terms of the deadline, the length of tasks, etc., the virtual machines that accept them need to be flexible to maintain the Quality-of-Service. This requires VM allocation mechanisms to be adaptive to the current state of the system. Our approach is to determine the dynamic nature of incoming virtual machines with their requests and use this to allocate them to hosts considering their resource request on hosts.

The input variables are a) requested CPU (CPU_R), b) requested memory (RAM_U), and c) requested bandwidth (BWD_U) of a virtual machine. The output variables in this controller are the a) CPU request weight (CPU_W), b) memory request weight (RAM_W), and c) bandwidth request weight (BWD_W). The membership function used in classifying VMs are shown in Figure 4 and are the linguistic terms LOW, MEDIUM, and HIGH are used. 27 rules with different states of inputs and outputs were defined in the Fuzzy Reasoning Engine. Once a rule is chosen that matches the Fuzzy sets, the next step is to defuzzify the output of said rule to obtain the weights. In the defuzzification, each output is plotted to a trapezoidal membership function and using the Center of Gravity (COG), the crisp output is obtained.

Algorithm 1 shows how each virtual machine is allocated to the suitable host. The Sort() method inputs the available hosts and the virtual machine's highest weight (CPU, RAM, BW), and based on the highest weight it will sort all the hosts, from lowest to highest based on utilization. For example, a virtual machine with CPU as the highest weight needs a host with a low utilization weight on CPU. If the virtual machine's weight values are the same, a prioritization of CPU-RAM-Bandwidth is established. The method can SupportVM() method checks if the host has enough resources for the virtual machine to use. Once a virtual machine is allocated to an available host, it will proceed to the next VM, else if all of the available hosts can't be allocated, it will select a new host. The weight will determine which type of resource each virtual machine will occupy on the host. This will pair virtual machines with their respective requests to hosts that will accommodate them according to the highest weight on the virtual machine's resource request.

```
Algorithm: Host Selection for VM Allocation
Input: HostList, VMList
For each vm in VMList
|  Sort(HostList, vm.HighestWeight)
|    For each host in HostList
|  |    If host canSupportVM(vm)
|  |  |    Then allocateHost(host, vm)
|  |  |     proceed to next VM
|  |    Else Proceed to next host
|    End For each host
End For each vm
```

(Algorithm 1) Host Selection for VM Allocation

3.2.3 Dynamic Upper Threshold using Utilization Weights of Hosts

To minimize the number of active hosts, this paper proposes the use of a dynamic upper threshold which utilizes the weights of hosts which is obtained from the output of the FIS on host's resource utilization in Section 3.2.1. Dynamic thresholds are implied in this proposal because using static thresholds in a dynamic cloud environment are unsuitable, especially for upper thresholds that are used in detecting overloaded hosts. Using these weights, the upper threshold is calculated based on the average weights found in Equation 1. CPU_w,RAM_w, and BW_w correspond to the weights of each resource $host_i$. VM_{cpu} is the number of virtual machines which CPU is considered to be the highest weight, VM_{ram} is the number of virtual machines in which RAM is their highest weight, and VM_{bw} is the number of virtual machines with BW as their highest weight. N_{vms} is the total number of virtual machines allocated to $host_i$. The lower utilization threshold of hosts is set to 30% (0.3), using this value as the low threshold is effective [10].

$$upThresh = \frac{(CPU_w * VM_{cpu}) + (RAM_w * VM_{ram}) + (Bw_w * VM_{bw})}{N_{vms}} \quad (1)$$

```
Algorithm: Host Selection for Over/Under utilized hosts and VM selection
Input: HostList, Output: migrationList
For each host in HostList
|  candidateList -> list of VMs to be removed from host
|  host Weights – get CPU, RAM, and BW weights from Fuzzy
|  If host Weights> host upperThresh
|  |  vmList -> retrieve and sort vm from host according to highest weight
|  |  For each vm in vmList
|  |  |  host Weights -> calculate when vm is removed
|  |  |  If host Weights > host upperThresh
|  |  |  |  vm -> add to candidateList
|  |  |  |  Continue to next vm
|  |  |  Else
|  |  |  candidateList -> add to migrationList
|  |  |  Proceed to next host
|  |  End For each
|  Else
|  If host Weights < host lowerThresh
|  |  All VMs of host -> add to migrationList
|  |  host -> go to sleep mode
|  Else host -> "STABLE"
End For each
Return migrationList
```

(Algorithm 2) Host Selection using the upper and lower thresholds with VM Selection

Algorithm 2 shows how the over and under utilized hosts are selected, along with the VM selection to decrease the overall utilization of the host. The weights of the host are compared to the *upperThresh* value. If the weights are over the *upperThresh*, then it will select which of its virtual machines must be migrated to reduce the utilization. The virtual machines selected in the process depend on the highest weight of resource utilization of the host.

The VM list *vmList* retrieves virtual machines that contributed to the highest resource utilization, if the highest weight of the host's utilization is CPU, then the list of VMs to be migrated are the ones classified as "CPU".

An estimated weight for the host is will replace the current weights of the host once the VM has been removed. The VM selection process will continue until all the three weights in $host_i$ are below the *upperThresh* value. Once all of the virtual machines have been selected, it is removed from the host and added to the migrationList for the VM Allocation process. Whenever a host's weight on resource utilization is below the *lowerThresh* value, if the weights are below the lower threshold, the algorithm will migrate all of the VMs from that host and change the status of the host to sleep mode to minimize the number of active hosts; else the host is considered to be stable, and therefore be available for accepting migrated VMs.

4. Simulation and Evaluation of Results

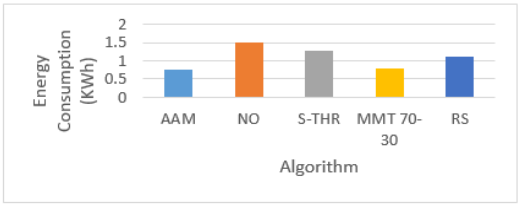
4.1 Simulation Environment

Virtual machines are implemented in an Infrastructure as a Service (IaaS) data centers. Therefore, we need to evaluate the proposed mechanisms on VM allocation and migration on a large scale virtualized data center infrastructure. CloudSim toolkit [11] was chosen to evaluate the proposed policies. CloudSim has the components to simulate cloud system components such as data centers and VMs and it supports policies for VM allocation and selection. For the implementation of the Fuzzy Logic System, we used jFuzzyLogic [12].

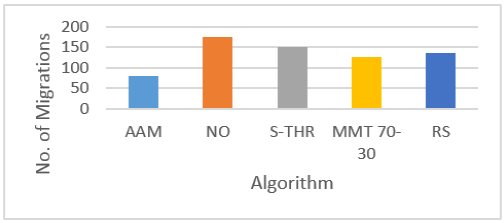
For the parameters in the CloudSim environment, a data center with 100 nodes was simulated with heterogeneous servers and VMs. Half of the servers are HP ProLiant ML110 G4, and the other half are HP ProLiant ML110 G5, with a CPU of 3000 MIPS, a memory of 15 GB and Bandwidth of 100 Mbps. We use the same power models provided on the web site [13] for both servers to calculate the energy consumption. For the dynamic properties of the virtual machines, each resource it needs is randomly generated, with the CPU, memory, and Bandwidth maximum values of 750 MIPS, 1024 Mb, and 4000 Kbps respectively.

4.2 Evaluation Results

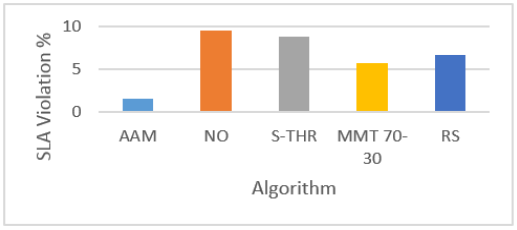
We evaluate the performance of our approach and compare its performance to other methods. These methods are a) No-limit (NO) approach which allocates virtual machines to hosts until the utilization of resources reaches 100%, then proceeds to another host, b) Single Threshold-based approach (S-THR) which sets an upper limit for host utilization and keeping the utilization below that limit which is calculated based on the average utilization of the VMs in the host, c) Double Threshold (70%-30%) with Minimum Migration Time (MMT), which sets the upper threshold to 70% and lower threshold of 30%, using MMT in choosing for virtual machines to migrate, and d) Double Threshold (70%-30%) and using Random Selection (RS) of virtual machines for migration. The evaluation metrics are the energy consumption, the number of migrations and SLA violation rate of each method.



(Figure 5) Total Energy Consumption



(Figure 6) Number of Migrations occurred



(Figure 7) Overall SLA Violation

Figure 5 shows the energy consumption of the respective strategies. The highest energy consumption is NO, with the value almost 1.2 KWh, followed by S-THR, and RS and MMT respectively. Our approach, Adaptive Allocation and Migration, AAM which consumed 0.75 kWh, is close to MMT and has the least energy consumption among other approaches. Therefore, using a dynamic threshold is efficient in the virtual machine migration, and using resource weights in virtual machine allocation is more effective compared than other approaches. This confirms that using the proposed migration mechanism in AAM reduces the number of active hosts.

The results in Figure 6 show that AVMA obtained the least number of migrated virtual machines, with a value of 80, followed by MMT, RS, S-THR, and NO with values of total migrations greater than 100. Therefore, utilizing dynamic thresholds in virtual machine migration is efficient because it chooses virtual machines that will balance the overall resource utilization of hosts. This shows that our approach has a more optimized mechanism for virtual machine migration where the over-utilization of resources among hosts is significantly reduced.

The results in Figure 7 show that AAM has the lowest overall SLA violation rate at 1.6%, followed by MMT with a rate of 5.7, RS with 6.7. Both S-THR and NO have higher SLA rates compared to others, with values of 8.79 and 9.49 respectively. Prioritizing the virtual machine's highest resource request allows the allocation process to assign virtual machines to hosts with the least available resource. This means that virtual machines will have a higher chance of meeting their SLA, compared to other approaches.

5. Conclusion

This paper presents an adaptive virtual machine allocation approach which utilizes Fuzzy Logic to determine the weights on the host's resource utilization and virtual machine's request on resources. Virtual machines with a high weight on a specific resource will be allocated to a host with the lowest weight on resource utilization. Using the weights on the host's utilization, a dynamic upper threshold was created to determine over-utilized hosts and to select virtual machines that will maintain the balance between all resource utilization. Virtual machines which contribute to the high utilization of the host's resource become the candidate for migration. Evaluation results show that our approach provides significant performance in terms of energy consumption, the number of migrated virtual machines, and SLA violation rate. In future works, a dynamic threshold can be applied in detecting under-loaded hosts.

참고문헌(Reference)

[1] T. Dillon, C. Wu, E. Chang, "Cloud computing: issues and challenges", Advanced Information Networking and Applications (AINA), 24th IEEE International Conference, 2010. https://doi.org/10.1109/AINA.2010.187

[2] J. Zhang, H. Huang, X. Wang, "Resource provision algorithms in cloud computing: A survey", Journal of Network and Computer Applications, vol. 64, pp. 23-42,

2016. https://doi.org/10.1016/j.jnca.2015.12.018

[3] R. K. Gupta, R. K. Pateriya, "Energy Efficient Virtual Machine Placement Approach for Balanced Resource Utilization in Cloud Environment", International Journal of Cloud-Computing and Super-Computing, vol. 2, no. 1, pp. 9-20, 2015.
http://dx.doi.org/10.21742/ijcs.2015.2.1.02

[4] M. Shelar, S. Sane, V. Kharat, R. Jadhav, "Efficient Virtual Machine Placement with Energy Saving in Cloud Data Center", International Journal of Cloud-Computing and Super-Computing, vol. 1, no. 1, pp. 15-26, 2014.
http://dx.doi.org/10.21742/ijcs.2014.1.1.02

[5] A. Beloglazov, J. Abawajy, R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for Cloud Computing", Future Generation Computer Systems, vol. 28, Issue 5, pp. 755-768, 2012..
https://doi.org/10.1016/j.future.2011.04.017

[6] D. A. Alboaneen, B. Pranggono, H. Tianfield, "Energy-aware Virtual Machine Consolidation for Cloud Data Centers", IEEE/ACM 7th International Conference on Utility and Cloud Computing, 2014.
https://doi.org/10.1109/UCC.2014.166

[7] A. Beloglazov, R. Buyya, "Adaptive Threshold-Based Approach for Energy-Efficient Consolidation of Virtual Machines in Cloud Data Centers", Proceedings of the 8th International Workshop on Middleware for Grids, Cloud, and e-Science, 2010.
https://doi.org/10.1145/1890799.1890803

[8] F. I. Elijorde, J. Lee, "Performance and Energy Oriented Resource Provisioning in Cloud Systems Based on Dynamic Thresholds and Host Reputation", Journal of Korean Society for Internet Information, Vol. 14, No. 5, pp. 39-48, 2013.
https://doi.org/10.7472/jksii.2013.14.5.39

[9] Y. Bai, D. Wang, "Fundamentals of Fuzzy Logic Control-Fuzzy Sets, Fuzzy Rules and Defuzzification", Advanced Fuzzy Logic Technologies in Industrial Applications, 2006.

[10] A. Beloglazov, R. Buyya, "Energy Efficient Resource Management in Virtualized Cloud Data Centers", 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, 2010.
https://doi.org/10.1109/CCGRID.2010.46

[11] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. Rose, R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms." Software: Practice and Experience, Wiley Press, NY, USA, 2010.
https://doi.org/10.1002/spe.995

[12] P. Cingolano, J. Alcala-Fdez, "jFuzzyLogic: A Robust and Flexible Fuzzy-Logic Inference System Language Implementation" 2012 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2012.
https://doi.org/10.1109/FUZZ-IEEE.2012.6251215

[13] All Published SPECpower_ssj2008 Results, http://www.spec.org/power_ssj2008/results/power_ssj2008.html

◐ 저 자 소 개 ◑

존 크리스토퍼 마테오 (John Cristopher A. Mateo)

2015년 West Visayas State University, Philippines BS in Information Technology
2015년~현재 Kunsan National University, South Korea, Graduate Student in Master's Course
관심분야 : Cloud Computing, Mobile Cloud Computing, Fuzzy Logic
E-mail : jcmateo@kunsan.ac.kr

이 재 완(Jaewan Lee)

1984년 중앙대학교 이학사-전자계산학
1987년 중앙대학교 이학석사-전자계산학
1992년 중앙대학교 공학박사-전자계산학
1996년 3월~1998년 1월 한국학술진흥재단 전문위원
1992년~현재 군산대학교 교수
관심분야 : 분산 시스템, 운영체제, 유비쿼터스 시스템, 클라우드 컴퓨팅 등, 모바일 클라우드 컴퓨팅
E-mail: jwlee@kunsan.ac.kr