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

모바일 클라우드 컴퓨팅에서 데이터센터 클러스터링과 가상기계 이주를 이용한 동적 태스크 분배방법☆

A Dynamic Task Distribution approach using Clustering of Data Centers and Virtual Machine Migration in Mobile Cloud Computing

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

요 약

모바일 기기로부터 클라우드 서버로 태스크를 오프로딩하는 방법은 클라우드랫(cloudlet)의 도입으로 인해 향상되었다. 동적 오프로딩 알고리즘을 통해 모바일 장비는 수행할 타스크에 적절한 서버를 선택할 수 있다. 하지만 현재의 태스크 분배 방식은 의사결정에서 중요한 VM의 수를 고려하지 않고 있다. 본 논문은 클러스터된 데이터 센터에서 동적인 타스크 분배 방법을 제시한다. 또한 서버에서 자원의 과부하를 방지하기 위해 할당된 CPU에 따라 VM을 균형있게 클라우드 서버에 이주시키는 VM이주 기법을 제안한다. 클라우드 서버의 이주 방법을 향상시키기 위해 최대 CPU 관점에서 데이터 센터의 자원 용량도 고려한다. 시뮬레이션 결과, 제시한 태스크 분배 기법이 전반적으로 시스템의 성능을 향상시켰음을 나타내었다.

☞ 주제어 : 클라우드 컴퓨팅, 클라우드렛, 모바일 클라우드 컴퓨팅, K-Means 클러스터링

ABSTRACT

Offloading tasks from mobile devices to available cloud servers were improved since the introduction of the cloudlet. With the implementation of dynamic offloading algorithms, mobile devices can choose the appropriate server for the set of tasks. However, current task distribution approaches do not consider the number of VM, which can be a critical factor in the decision making. This paper proposes a dynamic task distribution on clustered data centers. A proportional VM migration approach is also proposed, where it migrates virtual machines to the cloud servers proportionally according to their allocated CPU, in order to prevent overloading of resources in servers. Moreover, we included the resource capacity of each data center in terms of the maximum CPU in order to improve the migration approach in cloud servers. Simulation results show that the proposed mechanism for task distribution greatly improves the overall performance of the system.

☞ keyword : Cloud Computing, Cloudlet, Mobile Cloud Computing, K-Means Clustering

1. INTRODUCTION

With the introduction of the cloudlet [1], offloading approaches in mobile applications can obtain real-time response and results. Providing the devices with one-hop, low latency connections, cloudlets can preprocess tasks or offload them to other cloud servers through a high-speed internet connection. In [2], Cloudlets can host virtual machines (VMs) that can accommodate tasks, but compared to the cloud data center, in terms of processing and storage, the cloudlet can accommodate lesser tasks. Offloading tasks from the mobile devices to the cloud directly poses some problems in terms of network latency issues. With the presence of the cloudlet, mobile devices can pass these tasks on it, which can do the initial process. In the facial recognition process [3], the cloudlet can handle the detection of faces, while the job of the cloud servers is to do the facial recognition. Cloudlet locations are established near users, making it more convenient for them to access such services. Choosing the proper data center for obtaining the minimal response of the application is one of the problems being pursued in the Mobile Cloud Computing

[1] [2] Dept. of Information and Communication Engineering, Kunsan National University, Jeollabuk-do, 573-701, Korea
* Corresponding author(jwlee@kunsan.ac.kr)
[Received 17 July 2016, Reviewed 26 August 2016, Accepted 02 October 2016]
☆ Following are results of study on the "Leaders in INdustry-university Cooperation" Project, supported by the Ministry of Education

area, and this paper tackles that problem.

This paper presents a way to improve the task distribution approach in mobile-cloudlet-cloud architecture through data center clustering and a VM migration mechanism that prevents overloading in data centers. The objectives of the proposed architecture are as follows:

- To implement a clustering method for data centers
- To provide data centers with a VM migration method, that enables it to detect and migrate virtual machines in order to decrease the resource consumption.
- To provide fast response in the task selection by selecting available data centers.

The proposed approach for the task distribution determines the network throughput of each data center from the cloudlet and the number of tasks assigned to the virtual machines. In this way, the decision making can prevent tasks from making a long queue, as it considers the number of virtual machines with a task assigned to it. The VM migration strategy uses a proportional allocation method on resource consumption of virtual machines. After determining which data centers are overloaded with virtual machines, it runs the algorithm of selecting virtual machines for the migration process.

2. RELATED WORKS

The MOCHA architecture in [3-5] consists of mobile devices, cloudlet and the cloud servers where mobile devices are connected to the cloudlet, instead to the cloud directly. In [3], the benefits of this architecture were implemented on a real-time face recognition system, called Cloud-Vision. The face detection is processed in the cloudlet, and each face detected was distributed to the cloud servers to compare faces in the database. The fixed and greedy approach in task distribution was introduced in the MOCHA architecture. The fixed approach, which distributes tasks accordingly to the servers, and the greedy approach, which distributes tasks to the servers with respect to the known response time of servers. In [4], the said architecture and approach were implemented on battlefield applications. Cloudlets are proposed to be deployed in military vehicles, with satellites as the medium in order to access servers. In [6], the basis of selecting cloud servers is the parameter of the network latency, and sending the same task on each cloud server was also implemented.

Allocation of the resources in the cloud is important in having a robust system and providing a high quality of service to users. With proper management of these resources, cloud servers can minimize or sometimes prevent overloading of the servers due to improper allocation of said resources. In [7], the approach determines servers which are overloaded with cloud services assigned and re-assigning these cloud services to servers with less load in proportions. Some of the methods and approaches were implemented as the virtual machine allocation in this paper. In [8], the artificial neural network (ANN), was used to predict the resource utilization of virtual machines in hosts and determine the overloaded hosts. By using the predicted results of virtual machines, the approach selects virtual machines that are good candidates for the migration process. In [9], the resources allocated to virtual machines, RAM, the number of CPU cores, and architecture type that will determine the VM migration strategy on physical hosts. In [10], virtual machines are allocated to servers according to their location. The K-means clustering was used in classifying and clustering these VMs before allocating them to servers.

3. SYSTEM MODEL AND METHODS

3.1 PROPOSED ARCHITECTURE

Figure 1 describes the proposed architecture. Applications on mobile devices with offloading decision techniques relay the tasks to the cloudlet. The mobile device is connected to the cloudlet through a WLAN connection. The cloudlet receives tasks from mobile devices then it decides which available data centers the tasks should be offloaded. After making the decision, it will offload tasks to the assigned data centers. The descriptions of each component with their respective modules found in the proposed architecture are as follows:

A) Mobile Device Layer

- Intelligent Offloading Decision Maker – a decision engine implemented in the application that determines

tasks should be offloaded or remotely executed.

B) Cloudlet Layer

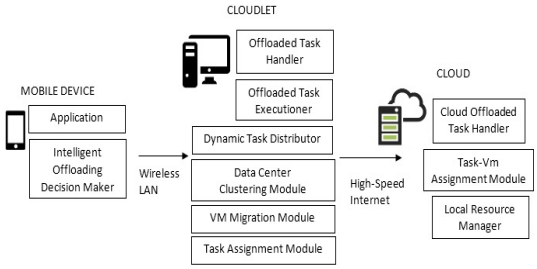
- Offloaded Task Handler – handles the tasks obtained from the mobile device.
- Offloaded Task Executioner – executes the tasks assigned to the cloudlet.
- Dynamic Task Distribution Module – handles the tasks that are assigned to the cloud.

(a) Data Center (DC) Clustering Module – module that clusters available data centers by their location.
VM Migration Module – this module implements a VM allocation to the selected cluster of data centers, where the virtual machines are properly allocated in order to balance the server's CPU utilization of all servers in that cluster.

(b) Task Assignment Module – after assigning the cluster of data centers, the cloudlet's task assignment module will decide on which servers in the clusters are the tasks will be assigned to. This will depend on the server's network parameters and the server load.

C) Cloud Layer

- Cloud Offloaded Task Handler – module that handles the tasks obtained from the cloudlet.
- Task-to-VM Assignment Module – module that handles the virtual machine that will accommodate the set of tasks.



(Figure 1) The proposed system architecture

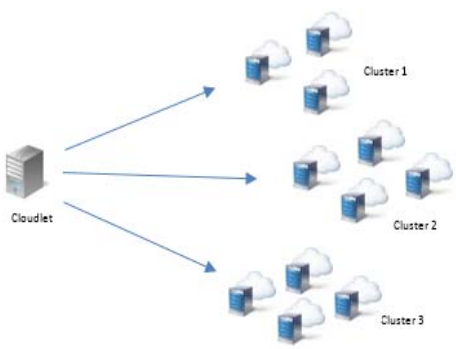
3.2 DATA CENTER CLUSTERING

K-means clustering [11] divides similar objects into groups, depending on the assigned value of K. In each K there is a cluster center or centroid that determines whether the points are included in the cluster by measuring the Euclidean distance of that point to all K-number of centroids. The cloudlet uses K-means clustering in order to divide the available data centers into clusters. The procedural steps are:

1. Specify the number of K clusters.
2. According to the value of K, calculate the value of the centroids by getting the average value of the points in the cluster. During the initialization phase, the centroids can be chosen randomly.
3. Proceed to calculate the distance of every point to the K centroids. The point will then be assigned to the closest cluster.
4. Recalculate the value of the centroid for each cluster.
5. Repeat step 3 until all the centroids do not change.

The values to be assessed during the clustering are the data center's X and Y values plotted in a 2-dimensional space. The value of K in determining the number of clusters to be applied is by using the Rule of Thumb, while the calculation of Euclidean distance is used to determine which points are nearest to a cluster. After the clusters have been determined, the tasks are evenly distributed to each cluster based on the number of clusters.

Figure 2 illustrates each of the clustered data centers, with a number of data centers. After the cloudlet clusters all of the data centers, it divides the tasks according to the number of clusters, then distributes it evenly. The proposed approach on VM migration is discussed next. The data centers are clustered according to so that the migration overhead in the VM allocation and migration strategy will be minimal.



(Figure 2) Data Center clustering method

3.3 VM MIGRATION STRATEGY

The VM migration strategy proposed in this paper was based on [7]. It was discussed that nodes are servers hosting the cloud services. Nodes are hosts in data centers, hosting virtual machines, and virtual machines that reside within each data centers are cloud services. The strategy is to balance the number of virtual machines in each data center in order to minimize servers with overloading virtual machines.

The idea of using Equations 1 and 2 where it sums up the resource utilization of cloud services in one node, and then the average resource utilization of all nodes. This threshold value determines on how many services with a specific resource consumption should be allocated to a node, in order to balance all of the nodes services' resource consumption.

$$\text{CPUDC} = \sum_{n=1}^{N} \text{CPU}\,\text{VMi} \quad (1)$$

$$\text{CPUmean} = \frac{1}{N}\sum_{n=1}^{N} \text{CPU}\,\text{DCi} \quad (2)$$

The first thing to do is to determine each VM resource consumption/utilization in the each host. Each virtual machine's allocated CPU resource in the data center is summed up, in Equation 1. After getting the sum of all the virtual machines in each data center, the mean CPU resource utilization, in Equation 2, in each data center, this value will be used in determining overloaded data centers, and distribute virtual machines to less loaded servers.

```
Output: DCOList, DCUList
1. For each dc in Data Center
2.   If CPU_DC >CPU_mean
3.     Then add DCOList.add(dc)
4.   Else
5.     If CPU_max - CPU_DC > CPU_mean
6.     Then DCUList.add(dc)
7.   Else Skip dc
8. End Foreach
```

(Figure 3) Underutilized and Overutilized CPU Detection in Data Center

The algorithm in Figure 3, the mean CPU utilization, CPUmean, was used in order to detect whether it is in the overload state when the current CPU utilization of the data center is greater than the mean value. The difference with this approach to the preferred approach is that it considers the maximum amount of CPU, CPUmax, which can be allocated by a data center. With the current CPU utilization deducted from the maximum CPU utilization of the data center, we can assume that the data center can accommodate other virtual machines. If the CPU utilization, CPUDC, is higher than the CPUmean value, it means that it can be a candidate for servers that will accept migrated virtual machines in order to lessen the load of others. These data centers are added to the Datacenter Overloaded list, DCOList. Meanwhile, servers with the current free CPU resources that are less than the CPUmean will go undergo another process. It will check if a specific server is available for accepting migrated virtual machines. Based on the value which is the maximum CPU allocated, CPUmax, minus the current CPU utilization of the of the data center, CPUDC, it will compare it to the threshold value, which is the CPUmean. If the result is greater than CPUmean, then that data center is available for accepting migrated virtual machines. Other servers not satisfying this condition will be skipped.

In Figure 4, the algorithm for virtual machine selection for migration is processed to the DCOList, which contains servers that have a higher CPU utilization of virtual machines compared to the mean value. First, the virtual machines in the data center are sorted from lowest to the highest value.

```
Input: VM_candidateList
1. For each dc in DataCenter
2.   For each vm in VM_candidateList
3.     CPU_DC = CPU_DC+ Utilization(vm)
4.     If CPU_DC< U_rc
5.     Then Allocate (vm to dc)
6.   Else stop
7.   End for each
8. End foreach
```

(Figure 4) VM Selection in Data Center for Migration Process

In each, the value of current utilization of the virtual machine, VMu, is deducted from the current CPU utilization of the data center. While the value of CPUDC is greater than

the CPUmean, the following virtual machine will be added to the candidates for migration. If this value is less than the mean CPU utilization of all data centers, the VM selection stops, then proceeds to the next data center in the DCUList.

```
Input: DCOList
Output: VM_candidateList
  1. For each dc in DCOList
  2.   Sort(vm,VM_u)
  3.     For each vm in dc
  4.       CPU_DC = CPU_DC - VM_u
  5.       If CPU_DC>CPU_mean
  6.       Then VMcandidateList.add(vm)
  7.       Else stop
  8.     End for each
  9. End foreach
```

(Figure 5) Proportional Allocation of VM to Data Center

In the algorithm in Figure 5, the chosen virtual machines for migration are then decided to which server it will be chosen migration, and how many of the candidates can be migrated in a single data center. The host on DCUList, or host in which their CPU utilization is less than the mean CPU utilization, is used in this algorithm. For each step the CPU utilization value of virtual machines will be deducted to the current utilization of data center. It will continue to add virtual machines according to the CPU utilization value until the CPU utilization for the data center does not exceed the CPUmean.

3.4 DYNAMIC TASK ASSIGNMENT TO CLOUD DATA CENTERS

After allocating the virtual machines to the designated servers, the cloudlet will now assign the tasks to the data center in the cluster, rather than cloudlets navigating on all of the available cloud servers. The server's parameters are used to determine the most efficient distribution of the tasks. The conventional approach does not consider the load of the server to which task is assigned, in this case, the number of busy and available virtual machines. Because of that, we considered the server load. But first, each cloud data center must be categorized first, by the following parameters:

- Network Throughput (NT)
  - Round Trip Time to from cloudlet to each data center, in seconds + Data transfer rate (File size required in the task/ network bandwidth) – KB / Kbps
- VM Usage
  - *VMused / VMmax*, if value is equal to 1 then VM usage in the server is full, or 100%
- Server Load
  - Total time of execution of all tasks assigned in server, measured in seconds

```
Input TaskList, ServerList
  1. for i = number of Tasks
  2. RankLowestHighest(ServerList, NT + SL_mean)
  3.   if Server VM usage = 1
  4.   then Skip/ Proceed to next Available
  5. else
  6.   Assign task_i  selected Server
  7.    Selected Server addVMused +1
  8. end for
```

(Figure 6) Proportional Allocation of VM to Data Center

For the task assignment algorithm, the servers from the *ServerList* are ranked with increasing order according to the Network Throughput (NT) + the average server load. The *ServerList* contains the servers located in one cluster. The server, obtaining the lowest sum of the two factors is chosen to be the appropriate server for the task. As the tasks are assigned to a cloud server, the amount of VMs used is counted. If the server has all of the VMs assigned to the tasks, it will be skipped during the next task assignment. This is to ensure that the tasks will not go on a queuing process that is waiting for other tasks to finish in the virtual machine before it can be processed. If there are remaining tasks and all of the virtual machines in the data centers are occupied, then the algorithm will allow the remaining tasks to proceed to queue. When there are tasks remaining during the distribution and all of the virtual machines in the data centers have been assigned, then the procedure will start again, using the remaining tasks, and enabling them to be assigned properly in data centers where

queuing time is present. The flow of the task distribution is found in Figure 6.

4. SIMULATION AND EVALUATION RESULTS

4.1 SIMULATION SETUP

Data center clustering is first performed on the simulation. 20 data centers with different hardware specifications and locations, given by X and Y coordinates, which are randomly created were used in the clustering method. After the clustering is done, we selected one data center, which is Cluster 1, in order to apply the proposed approach to VM allocation and task distribution. The simulation was performed in the CloudSim toolkit [12], which is a Java-based framework imported on the Eclipse Juno IDE. After the clustering was finished, we chose the first cluster, then applied the proposed VM allocation/migration and task distribution strategy. Table 1 shows the data centers included in Cluster 1 during the simulation, and this cluster was used in the task distribution approach.

(Table 1) 10 out of 20 Data Center with their properties

Data Center	MIPS	No. of Cores	Memory (MB)	X, Y	Cluster
Data Center 1	3300	16	10240	640, 422	1
Data Center 2	2900	16	10240	443, 193	1
Data Center 3	3000	16	16384	742, 285	1
Data Center 4	2700	16	12288	486, 423	1
Data Center 5	3200	16	10240	349, 752	4
Data Center 6	3400	16	2288	805, 589	2
Data Center 7	3000	16	10240	624, 719	4
Data Center 8	2700	16	2288	914, 356	3
Data Center 9	2800	16	16384	967, 296	3
Data Center 10	3300	16	10240	832, 602	2

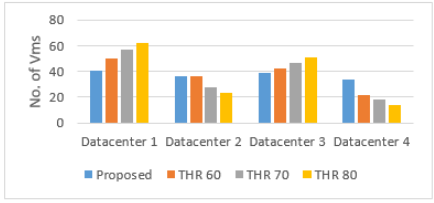
For the VM migration, the simulator environment consists of data centers found in Cluster 1, with its parameters (number of cores, millions of instruction it can process per second, MIPS, for each core, storage, memory, etc.), and 150 virtual machines with parameters (1000-2500 mips, 256-2048 Mb.) were randomly created.

For the task distribution approach, tasks were created with the parameters of task length (in million instructions) from 1000-2500, and the file input/output sizes were also randomly created, from 400-600 in kilobytes. The network parameters [13] [14] of each data center are as follows: 100 Mbps bandwidth and latency ranging from 0.2 – 10 seconds.

4.2 EVALUATION RESULTS

To determine whether the VM migration and task distribution contribute to the improvement of the Mobile-Cloudlet-Cloud architecture, the approach was compared to conventional approaches. The results were then compared to other approaches to determine its performance in the said architecture. To evaluate the efficiency of using the mean of the CPU utilization as a threshold, we compared it to the static single thresholds, 60%, 70% and 80% respectively.

For the virtual machine migration, the virtual machines were randomly allocated to the data centers initially, after that, it will compute for the current CPU utilization of each data center, and distribute VMs in each data center to balance CPU utilization. In Figure 7 shows the number of virtual machines reallocated to each data center after the migration. Virtual machines on all datacenters using the proposed threshold are balanced. Using static thresholds, the most number of virtual machines allocated are in data centers 1 and 3, leaving data centers 2 and 4 less allocated. This is because data centers 1 and 3 have the highest CPU resource, therefore the priority on allocation is high. But the number of virtual machines does not reflect its CPU utilization.

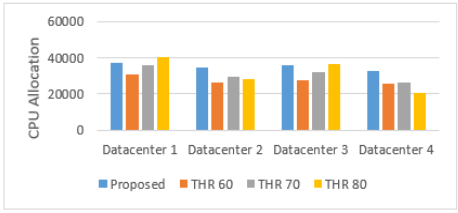


(Figure 7) Total number of virtual machines assigned in each data center

As shown in Figure 8, data Center 4 has the lowest total CPU allocated to virtual machines when using static thresholds. This is because data center 1 has the highest CPU capacity, following data center 3, meaning that most of the virtual machines are migrate to these data centers. Using static threshold, data centers 2 and 4 are under allocated. However,

the CPU allocation using the proposed threshold, data centers 2 and 4 utilization on CPU are the same with data centers 1 and 3.

For the task distribution, both Fixed and Greedy were the conventional approaches in the simulation, and the results will be compared to the proposed approach. Fixed Approach assigns an equal amount of tasks for each data center in the cluster, while Greedy approach considers the response time of the last task assigned to it, then chooses the data center with the least response time. Both of these approaches do not consider the number of virtual machines that each data center has hosted. In this case, this may lead to tasks going to queue more often, increasing the time for response to return. In the simulation, the number of virtual machines is as follows: Data center 1: 41; Data center 2: 39; Data center 3: 36; and Datacenter 4: 34, for a total of 150 virtual machines.

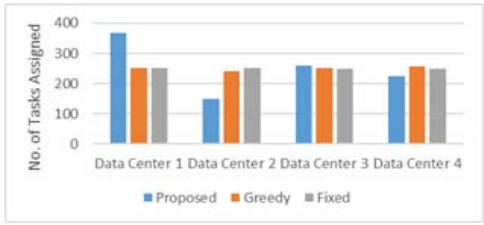


(Figure 8) Total amount of CPU allocated to each virtual machine in each data center

Figure 9 shows that in the proposed approach, most of the tasks are assigned to the Data Center 1, strictly because it has the most virtual machines compared to others. Notice that the proposed approach's effect on Data Center 2, because it holds the least virtual machines, the algorithm will assign fewer tasks with respect to it, as to prevent tasks from entering queuing time.

(Figure 9) Total number of tasks assigned to each data center

The Greedy and Fixed methods simply ignore this, it may not be seen here, but in Figure 10, we can see that the Greedy and Fixed approaches effect on Data Center 2. Both the Greedy and Fixed approaches, because both do not consider the number of virtual machines in a data center in assigning tasks, most of the tasks are not yet processed when they are received by a data center. This is because there is no virtual machine to accommodate it, leaving the tasks to wait for the first ones to arrive at finish first. The approach adjusts to the number of virtual machines that are present.



(Figure 10) Total roundtrip time of the tasks assigned in each data center.

5. CONCLUSION

The Mobile-Cloudlet-Cloud architecture was introduced to improve the performance of applications and services in the Mobile Cloud Computing field. In this paper, an improvement to the current approach in task distribution on MOCHA architecture is presented. In existing approaches, the greedy and fixed approaches did not include various limitations found in the cloud environment, specifically in data centers. Clustering of data centers with respect to their locations was implemented to minimize transmission delays between data centers. An approach to balancing the virtual machine's CPU utilization in each data center by utilizing an upper threshold then selecting virtual machines until the utilization is below the threshold proves to be much efficient when compared to using static thresholds. In the task distribution results, our approach shows a better performance compared to the conventional methods, by considering the amount of busy and available virtual machines. Future research can improve the task distribution on data center clusters by implementing other intelligent methods.

참 고 문 헌 (Reference)

[1] M. Satyanarayanan, P. Bahl, R. Caccres, N. Davies, “The Case for VM-Based Cloudlets in Mobile Computing”, IEEE Pervasive Computing, 2009. http://dx.doi.org/10.1109/MPRV.2009.82

[2] A. Bahtovski, M. Gusev, “Cloudlet Challenges”, 2th DAAAM International Symposium on Intelligent Manufacturing and Automation, 2013. http://dx.doi.org/10.1016/j.proeng.2014.03.045

[3] T. Soyata, M. Kwon, W. Heinzelman, “Cloud-Vision: Real-time Face Recognition using a Mobile-Cloudlet-Cloud Acceleration Architecture”. International Symposium on Computers and Communications, July 2012. http://dx.doi.org/10.1109/ISCC.2012.6249269

[4] T. Soyata, R. Muraleedharan, J. Langdon, C. Funai, S. Ames, M. Kwon, W. Heinzelman, ”COMBAT: mobile-Cloud-based cOmpute/coMmunications infrastructure for BATtlefield applications”, May 2012. http://dx.doi.org/10.1117/12.919146

[5] H. T. Mouftah, N. Kantarci, “Communication Infrastructure for Cloud Computing”, Advances in Systems Analysis, Software Engineering, and High Performance Computing (ASASEHPC). http://tolgasoyata.com/file/soyata.igi13.pdf

[6] M. Kwon, Z. Dou, W. Heinzelman, T. Soyata, H. Ba, J. Shi, “Use of Network Latency Profiling and Redundancy for Cloud Server Selection”. http://dx.doi.org/10.1109/CLOUD.2014.114

[7] R. M. A. Mateo, J. Lee, “Dynamic Service Assignment based on Proportional Ordering for the Adaptive Resource Management of Cloud Systems”, KSII Transactions on Internet and Information Systems, vol. 5, no. 12, December 2011. http://dx.doi.org/10.3837/tiis.2011.12.002

[8] F. I. Elijorde, J. Lee, “Optimizing Performance and Energy Efficiency in Cloud Data Centers Through SLA-Aware Consolidation of Virtualized Resources” Journal of the Korean Society for Internet Information, vol. 3, article 15, June 2014. http://dx.doi.org/10.3937/tiis.2011.12.002

[9] T. D. Nelsen, C. Iversen, P. Bonnet, “Private Cloud Configuration with MetaConfig”, IEEE 4th International Conference on Cloud Computing, 2011. http://dx.doi.org/10.1109/CLOUD.2011.63

[10] B. Panchal, R. K. Kappor, “Dynamic VM Allocation Algorithm using Clustering in Cloud Computing”, International Journal of Advance Research in Computer Science and Software Engineering, vol. 3, issue 9, September 2013. http://www.ijarcsse.com/docs/papers/Volume_3/9_September2013/V3I9-0119.pdf

[11] T. M. Kordinariya, P. R. Makwana, “Review on determining the number of Cluster in K-Means Clustering”, International Journal of Advance Research in Computer Science and Management Studies, vol. 1, issue 6, November 2013. http://academia.edu/5514429/Review_on_determining_number_of_Cluster_in_K-Means_Clustering

[12] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, R. Buyya, “CloudSim: a toolkit for modeling and simulation of Cloud computing environments and evaluation of resource provisioning algorithms”, Software: Practice and Experience, pp. 23‐50, 2011. http://dx.doi.org/10.1002/spe.995

[13] “3G, 4G and Wi-Fi Network bandwidth”, http://www.bandwidthplace.com/internet-speed-test-3g-4g-lte-and-wifi-who-wins-article/

[14] “3G, 4G and Wi-Fi Network Latency”, http://www.evdoinfo.com/content/view/4818/64/

◐ 저 자 소 개 ◑

존크리스토퍼테오 (John Cristopher A. Mateo)
2015년 West Visayas State University, Philippines
BS in Information Technology
2015～현재 Kunsan National University, South Korea, Graduate Student in Master's Course
관심분야: Cloud computing, mobile cloud computing, intelligent algorithms

이 재 완 (Jaewan Lee)
1984년 중앙대학교 이학사-전자계산학
1987년 중앙대학교 이학석사-전자계산학
1992년 중앙대학교 공학박사-전자계산학
1996년 3월~ 1998년 1월 한국학술진흥재단 전문위원
1992 ~ 현재 군산대학교 교수
관심분야 : 분산 시스템, 운영체제, 유비쿼터스 시스템, 클라우드 컴퓨팅 등