A Hybrid Cloud Resource Clustering Method using Analysis of Application Characteristics

Yoori Oh, Yoonhee Kim
Dept. of Computer Science, Sookmyung Women's University,
Seoul, Republic of Korea
Email: {yoori0203, yulan}@sookmyung.ac.kr

***Abstract*— With the development of cloud computing technology, there are many scientists who want to perform their experiments in cloud environments. Because of the pay-per-use method, it is cost-optimal for scientists to only pay for the cloud services needed for their experiments. However, selection of suitable resources is difficult because they are composed of various characteristics. Therefore, a method of classification is needed to effectively utilize cloud resources. Static classification of a resource can derive inaccurate results, while scientists submit various experiment intentions and requirements. Thus, a dynamic resource-clustering method is needed to accurately determine application characteristics and scientists' requirements. In this paper, a resource-clustering analysis, which considers application characteristics in a hybrid cloud environment is proposed. The resource clustering analysis applies a self-organizing map and the k-means algorithm to cluster similar resources dynamically. Performance is verified by comparing the proposed clustering method with other studies' resource classification methods. Results show that the proposed method can classify similar resource cluster reflecting application characteristics.**

Keywords— hybrid cloud, self-organizing map, cluster analysis, dynamic resource clustering

I. INTRODUCTION

In recent years, cloud computing technology has received much attention and praise by enabling cloud users to borrow virtual resources according to their needs and conveniences without time and space constraints (as long as Internet connections are available). There are various cloud service providers, including many cloud infrastructure providers (IaaS), such as Amazon EC2 [1], Microsoft Windows Azure [2], KT Ucloud [3], Google Cloud Engine [4], NHN Entertainment [5], and many others that provide cloud computing services. However, because of non-unified description methods of resource specification offered by those providers, the cloud user has to expend time and energy researching and comparing available resources in search of appropriate resources for him or herself. Therefore, a need exists for standardized notations of resource specifications to make it possible for cloud users to choose appropriate resources that meet their needs without the constraints of cloud computing service vendors in hybrid cloud environments, which offer intercompatibility between cloud services.

Because cloud resources are composed of a wide variety of characteristics, from the point of view of scientists who desire to carry out experiments with various scientific applications, it is not a simple matter to select the relevant cloud resources from the many available vendors, each of which may offer its own unique features and appealing offers. Therefore, classification of cloud resources is necessary to efficiently select and utilize resources. On the other hand, because scientists submit various intentions and requirements for their experiments according to the nature thereof, employing a static classification method to cloud resources might yield uncertain and vague results. In such a case, it is necessary to dynamically change the classification of cloud resources by varying the viewpoints according to the application characteristics depending on the situation.

In this research paper, a method that clusters resources in hybrid cloud environments is proposed. Through the suggested algorithm, the proposed method can form clusters of similar resources by reflecting application characteristics, through which efficient resource selection is possible.

The rest of this research paper is organized as follows: Section 2 introduces related research on classification methods of cloud resources that utilize various schemes. Section 3 specifies characteristics of virtual resources in a hybrid cloud-computing environment and discusses resource cluster analysis that accurately reflects application characteristics. Section 4 carries out experiments to evaluate the performance of the suggested algorithm and then analyzes the results of the experiments, and finally, section 5 concludes the research.

II. RELATED WORK

In this section, studies related to resource classification methods utilizing a wide variety of methods are introduced. In order to conveniently manage cloud resources of various properties and performances, it is necessary to summarize or classify them while considering the factors of those cloud resources. By employing machine learning and statistical methods, it is possible to summarize and classify a large amount of data. Thus, many studies are being carried out on the classification methods that apply clustering of resources [7, 8, 9, 10].

When clustering virtual cloud resources, Chavan [7] applies the k-means clustering method and forms clusters by

978-1-5090-6558-5/17 $31.00 © 2017 IEEE
DOI 10.1109/FAS-W.2017.162

considering the memory allocated to virtual resources. In the research, clustering is used in the reconstruction and scheduling of virtual resources. It guarantees high availability of resources with the enhanced scalability via the utilization of formed clusters.

Whaiduzzaman et al. [8] states that each cloud customer should be able to select the appropriate service according to his or her needs, and that there have been various approaches proposed to resolve service selection issues. Among those approaches, Multicriteria Decision Analysis (MCDA) enables a user to select from among many available cloud services. In the research, Whaiduzzaman et al. [8] classifies and synthesizes several MCDA techniques according to their categories, types, and methods, and explains the execution methods of each MCDA technique.

Wu and others [9] classify cloud service resources by employing the Bayesian classification algorithm. The research classifies cloud resources by computing several feature similarities, especially for certain factors that have greater effects on classification. It utilizes a Bayesian algorithm that introduces weights to compute feature similarity. However, the above research did not take applications into consideration, and only classifies resources of various types. In addition, its aim is to enhance the efficiency of classification via parallelization.

The research by Ahn Younsun et al. [10] is somewhat similar to this research, and they propose a dynamic resource classification method that considers application characteristics in Intercloud environments. By utilizing the k-modes algorithm, a statistical method, weights are assigned according to characteristics of applications, forming clusters. However, the k-modes algorithm computes similarity by identifying whether the values of the pertaining characteristics are the same. Since it can contribute to similarity only when the values of the properties are equal, it makes it difficult to reflect similar but not identical property values. Thus, the method is insufficient in identifying similarity between two resources.

III. Resource Clustering Method using Analysis of Application Characteristic

In this section, resource characteristics in hybrid cloud computing environments are specified, and a resource cluster analysis is introduced. The resource cluster analysis that reflects application characteristics proposes cluster derivation employing: a self-organizing map (SOM)-based application characteristics weight reflecting model learnings [12], and Elbow method-based k-means cluster analysis [13].

A. Resource Characteristic Specification

In order to integrate management of various resource notation methods of cloud computing service providers, characteristics of cloud resources are specified in this section. This research specifies cloud research characteristics by fully utilizing the resource characteristic specification employed in the research [10]. The research [10] employs and expands the mOSAIC ontology of the mOSAIC project [13] to specify consistent resource characteristics for various cloud resources.

In this research, we specify virtual resources by applying the definitions of resource-characteristics of private and public cloud infrastructure providers, such as OpenStack [6], Amazon EC2 [1], Microsoft Window Azure [2], KT Ucloud [3], Google Cloud Engine [4], and NHN Entertainment [5]. A total of 97 virtual cloud resources have been specified: 5 resources of OpenStack; 24, Amazon EC2; 25, Microsoft Window Azure; 18, KT Ucloud; 12, Google Cloud Engine; and 13 resources of NHN Entertainment. Drive type was additionally specified based on the resource characteristic specification of the research [10]. The resource information of Google Cloud Engine [4] and NHN Entertainment [5] was also added. Table 1 exhibits a portion from the resource specifications of the 6 cloud resource providers' 97 cloud resources. The first line of Table 1 lists the specified cloud resource characteristics. The characteristics of cloud resources are: names of cloud resources (resource name), names of cloud service providers or i.e., vendors (provider), memory capacity (Memory Size, GB), processing speed of virtual CPU (CPU flops), network bandwidth (network bandwidth), CPU (vCPU), cloud storage capacity (storage, GB), cost (unit: $0.01), and types of virtual drives (HDD or SSD). Starting from the 2nd line, Table 1 reflects the resource characteristic information of each cloud resource.

Table 1 Partial list of resources applying the resource characteristics specification

Resource Name	Provider	Memory size (GB)	CPU Flops	Network Bandwidth	vCPU	Storage(GB)	Cost (unit: $0.01)	Drive type
N-m2.small	NHN Entertainment	2	2.4	5	1	30	4	HDD
m4.xlarge	Amazon EC2	16	2.4	8	4	30	33.1	HDD

B. Resource Cluster Analysis that reflects Application Characteristics

Algorithm 1 clusters various cloud service providers and private cloud resources based on characteristics of applications. A set of application characteristics is a resource factor that affects the execution of an application, and composed of elements such as CPU, memory, and networks.

Application Characteristics AC = {CPU, memory, network, …} (1)

Algorithm 1. resource cluster analysis considering application characteristics

Application Characteristics *AC* = {CPU, Memory, Network, ... }

1) **Submit** Application *App*
2) **Identify** the Most Important Application Characteristics *AC*
3) **Set** *InputData* = {*AC*, Resource List *RL*}
4) Group of Output Nodes *SOM model* <- **SOM training**(*InputData*)
5) Cluster Lists *CL* <- **k-means clustering** (*K*, *SOM model*) based on **elbow method**.

The cloud user submits the application (App) to be executed (line 1). Search for the characteristics (AC) that have the greatest effects on the execution of the submitted application (line 2). Prepare the input data (InputData) for SOM trainings (line 3). The input data is made of application characteristics (AC) and resource list (RL) based on the Table 1 resource specification of the 3.1 resource characteristics specification.

$$\text{InputData} = \{\text{AC, Resource List RL}\} \quad (2)$$

Derive a SOM model by utilizing Self-Organizing Map training (SOM training) (line 4). By using the input data (InputData) as input neurons, derive a SOM model as output neuron group. Based on Elbow method, by employing k-means cluster analysis (k-means clustering), output the formed cluster results (CL) (line 5).

Resource cluster analysis that takes application characteristics into consideration is roughly divided into two steps to carry out the analysis procedure. The first step involves mappings of cloud resources to competing layers using SOM. At the time, the cluster was formed by assigning weights to characteristics that affect the execution of the application submitted by the cloud user. During the second step, k-means cluster analysis is performed on the results of the first step. At this time, the optimum cluster number k is determined employing the elbow method.

1) Training of model reflecting application characteristics weight using SOM

Algorithm 2 is a model learning algorithm using SOM [12]. Before starting the algorithm, in order to execute SOM[12], map size of output neurons should be determined. In addition, the number of iterations (Iter) for the training and application characteristic weight priority order (α) and learning rates (β) are set.

Initialize the current number of iterations to zero (line 1). In addition, initialize all of the connected weight vectors (W_{ij} (0)) to random values (line 2). Present a new input vector as the input neuron (line 3). The input vector is composed of the resource list (RL) (line 3 of algorithm 1), and after the normalization process, it is presented as the input neuron. Repeat the iteration until the maximum number of iterations is reached (line 4). N represents the number of properties of the input vector; per each property, derive the sum of squares of the distances between the input neurons (X_i (t)) and the connected weight vector (W_{ij} (t)) (line 5). At this point, K_i represents the weight for application characteristics (AC) (line 2 of algorithm 1), and it is derived by multiplying the factor that affects application execution by the weight. When the application submitted by cloud users has several factors to assign weights to, it can assign different weights according to the priority order (α). Select the winning neuron with the shortest distance between the derived distance squared sums (line 6). Update the weights associated with the neighboring output neurons of the winning neuron (line 7). The learning rate (β) has a value between 0 and 1 and determines the degree of updating of the weight to be connected. Increase the current number of iterations (line 8) and continue training by repeating the iteration until it reaches the determined maximum number of iterations.

Algorithm 2. training model reflecting application characteristic weight using self-organizing map

t = current iteration

Iter = max iteration number

$X_i(t)$ = i th input neuron at iteration t

$W_{ij}(t)$ = weight between i th input neuron and j th output neuron

1) t <- 0
2) $W_{ij}(0)$ <- random number, for all i and j
3) Input neuron <- input vector
4) **While**(t != Iter)
5) **Calculate** d_j for all output neuron

$$d_j = \sum_{i=0}^{N-1} k_i (X_i(t) - W_{ij}(t))^2, k_i = \begin{cases} 1, & i \neq characteristic \\ \alpha, & i = characteristic \end{cases}$$

6) **Select** j with minimum d_j
7) **Update** j and j's neighbor neuron's weight vector

$$W_{ij}(t+1) = W_{ij}(t) + \beta(X_i(t) - W_{ij}(t))$$

8) t++
9) **EndWhile**

2) Cluster derivation employing Elbow method-based k-means cluster analysis

Perform the k-means cluster analysis [13] using the learned model that exploits SOM [12] of step 1 mentioned above. Each output neuron of the resulting model trained in step 1 has the weight associated with it. Use this as the input value of k-means cluster analysis [13]. At this time, select the optimal K number of clusters by adding the number of clusters and executing the elbow method [14].

When proceeding with the k-means cluster analysis [13], in order to derive the optimal number of clusters, execute the

Elbow method [14]. The Elbow method [14] computes the squared sum (wcss; within cluster sum of squares) of the distance between the centroid of the cluster and the factor that belongs to the cluster, according to the number of clusters.

$$\text{wcss} = \Sigma \ \Sigma \ dist(x - ci)2 \ x \in Si \qquad (1)$$

Equation 1 means when the n number of data set (x_1, x_2, ..., x_n) is given, it sorts the data set into a K number of clusters S = { S_1, S_2, ..., S_k}. The ci represents centroid of the set S_i and the square sum of the data in the cluster is derived from the centroid of each cluster. As the number of clusters increases, the value of wcss decreases. When the wcss value does not decrease greatly, even with the addition of a number of clusters, the previous number of clusters is then selected as the optimum number of clusters in step 1.

IV. Experiment

In this section, to evaluate the algorithm suggested in this research paper, experiments were carried out and the results were analyzed. First, the target application characteristics are explained and the analysis results of cloud resource clusters are examined. Finally, the conducted experiments are explained and the results are analyzed.

A. Application Characteristics

In order to validate the superiority of the method proposed in this paper, experiments were conducted by employing two different scientific applications. Two applications used in the experiments were computational fluid analytical dynamics simulation (Computational fluid dynamics; CFD)[15], used in the field of aerospace, and the image mosaic engine (Montage GALFA)[16] of astronomy fields.

The experiments were executed by the two-dimensional Euler equation among CFDs. Grids of 2 KB were used, and the experiments were conducted by setting in the input file the target time as 0.5 and the maximum number of iterations to 1,000,001.

The Montage GALFA, an image mosaic engine that is used in the astronomy field, requires a large capacity for input data to generate a great amount of intermediate data and output data. In this research, experiments were conducted using 15 planes at the shrink stage.

In this research, the CFD application has CPU intensive application characteristics and the CPU is set as a resource factor affecting application execution. In addition, the Montage application is a memory-intensive application, and the memory is set as a resource factor that affects execution.

B. Result of Resource Cluster Analysis

When the application characteristic weight-based model training using SOM was performed in this research, 64 output neurons (8 * 8) were set and the weight of application characteristics was set to 2. In addition, the learning rate used to update the weight associated with the output neuron is set to 0.05 and linearly decreased to 0.01. The designated training number of iterations is set to 500. When the number of clusters increased during the k-means cluster analysis based on Elbow method was employed to derive clusters, the case where the difference between the previous number of clusters and the wcss value are less than 10 was selected as the ultimate number of clusters.

Table 2. Partial list of input neurons of model training reflecting application characteristics weights

resource name	Provider	Memory	Cpu flops	NW Bandwidth	vCPU	Storage	Cost	Drive type
Standard8	0.36	-0.07	-0.56	-0.48	0.56	0.07	0.28	-0.77
N-m2.xlarge	1.05	-0.07	-0.56	-0.48	0.56	-0.70	0.07	-0.77

Table 2 shows a partial list of input neurons for training the model (using algorithm 2) reflecting the weights of application characteristics. Resources are normalized and stored as shown in Table 2. The cloud resource is mapped on output neurons via training iterations. If the application is not considered, then the entire resources in Table 2 are assigned to different neurons, and via the k-means cluster analysis, {Standard7, Standard8, N-m2.xlarge, n1-highcpu-8} is classified as cluster 1, {n1-standard-8, n1-highmem-8} as cluster 5, and {N-c2.large} is classified as cluster 6. If the CFD application is considered, {n1-standard-8, n1-highmem-8} is to be mapped on neuron 25, {n1-highcpu-8} on neuron 26, and {Standard7, Standard8, N-m2.xlarge, N-c2.large} will be mapped on neuron 27. In addition, through the k-means cluster analysis, all neurons are classified as one cluster, cluster 2. In this case, when compared with the cases not considering applications, by considering the characteristics of CFD application, resources with same vCPU values are clustered into one cluster. When Montage application characteristics are taken into consideration, {Standard7, Standard8} is mapped on neuron 13, {N-m2.xlarge, N-c2.large} on neuron 5, {n1-standard-8} on neuron 1, {n1-highmem-8} on neuron 9, and {n1-highcpu-8} is mapped as neuron 6. The {Standard7, Standard8, N-m2.xlarge, N-c2.large, n1-highcpu-8} is classified as cluster 2, {n1-standard-8} as cluster 1, and {n1-highmem-8} as cluster 3. Since the weight of the memory property is given, during the formation of clusters, the clusters are formed according to the similarity of memory values.

C. *Performance comparison with other cluster analysis methods*

With the Amazon EC2 [1]'s 3 resources (m4.xlarge, m4.2xlarge, m3.2xlarge) and Microsoft Window Azure [2]'s 2 resources (A3, F8), Openstack [6]'s 2 resource (m1.medium, m1.large), CFD application and Montage application were executed repeatedly to compare standard deviation of the execution time (i.e., run time, run duration).

Comparative experiments were conducted on preceding research [10] that is similar to this research, in which derived clusters of cloud resources reflect the characteristics of scientific applications. As for the resources in Table 1, the vCPU was set as one of the CFD application characteristics and then assigned weights. MemorySize was set as the Montage application characteristic. By applying the k-modes method proposed in the preceding research [10], clusters were formed. Among those resource clusters that were formed, group 1 inside the cluster that includes the resource used in the execution time similarity comparative experiments was set as the centroid, and the resource was used as a sample to measure the execution time. Clusters formed using methods proposed in this research and cluster results using methods of preceding research [10] are different. Therefore, the resource in cluster 1 that had been used in the execution time similarity comparative experiments and the group containing the most common resources are designated as similar groups.

Figure 1 shows the result of comparing the standard deviation of the cluster and that of the group for the similar group by measuring the execution time of the CFD application five times for each resource. The clustering method proposed in this research is called Som-based Clustering (SBC) and the clustering method proposed in the research [10] is called K-mode based Clustering (KBC). Group 4 is composed of m1.medium, m1.large, and m4.2xlarge resources; group 5 of m3.2xlarge, N-m2.xlarge, N-c2.large resources; and group 7 of A3 and F4. Cluster 2 and group 7 have A3 as their common resource, cluster 4 and group 4 have m1.medium and m1.large as their common resources, and cluster 5 and group 5 have m3.2xlarge as their common resource. Therefore, each cluster and group is set to correlate. The CFD application was executed on resources composed of each cluster and group in order to measure execution times, and their standard deviation was derived. From the results, it is possible to say that in all cases, when the SBC method is applied, the standard deviation inside clusters is small. The difference is greatest for the standard deviations of cluster 5 and group 5, which are 16, and 20.2, respectively.

Figure 2 exhibits the comparative experimental results of the standard deviation between the proposed method and the previous research [10]. The experiment results were derived by repeatedly executing the Montage application five times for each resource. Group 2 is composed of m4.xlarge, N-c2.xlarge, and D13 resources; group 3 contains m4.2xlarge and N-r2.small resources; and group 4 is made of m1.medium and m1.large resources. Cluster 5 and group 3 have m4.2xlarge, cluster 6 and group 2 have m4.xlarge, and cluster 7 and group 4 have m1.medium and m1.large as their common resources. Thus, each cluster and group are set to correlate. By executing the Montage application with the resources that compose each cluster and group, execution times were measured, and their standard deviations were derived. In cases of cluster 5 and group 3, the difference between their standard deviations was 215, the greatest difference. As for the cluster 7 and group 4, the resources are the same, thus their standard deviation is the same at 96.

The results of measuring the application execution time for the resources constituting the similar cluster and group confirm that the standard deviation of the SBC proposed in this research is smaller than that of KBC proposed in preceding research [10]. By this result, it is possible to see that the proposed resource cluster analysis method forms accurate clusters for similar resources by reflecting application characteristics.

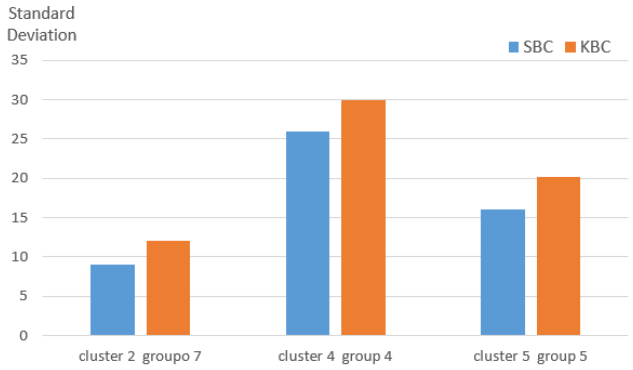

Figure 1 Standard deviation comparison of research reflecting CFD application characteristics

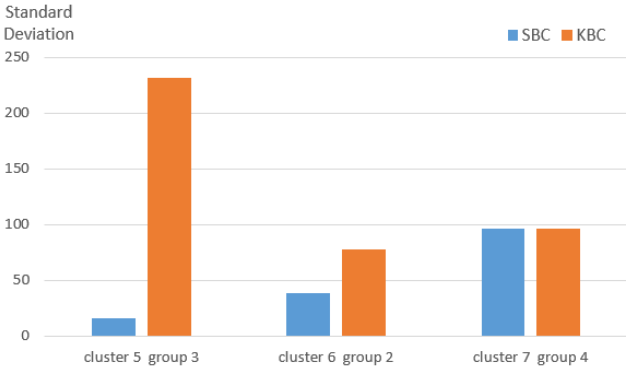

Figure 2 Standard deviation comparison of preceding research reflecting Montage application characteristics

V. CONCLUSION

This research demonstrated the usefulness of a dynamic resource cluster analysis method that reflects characteristics of applications in hybrid cloud environments. The proposed resource cluster analysis clusters similar cloud resources together by applying SOM and the k-means algorithm. Based on the proposed algorithm, applications have been executed using actual virtual resources from various existing cloud

service providers. In addition, compared with other cluster analysis methods, applications were executed by employing the dynamic resource classification method proposed in this research, and then by employing others, the results showed that the proposed method is excellent in comparison to other methods.

Future work will investigate ways to add and expand cloud resource environments and apply the further supplemented version of the proposed algorithm. In addition, to further our research, we intend to extend the investigation into efficient resource cluster analysis and recommendation algorithm for multi-characteristics by assigning various characteristics and priority orders to applications.

ACKNOWLEDGMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning (NRF-2017R1A2B4005681)

REFERENCES

[1] Amazon EC2, http://aws.amazon.com/

[2] Microsoft Window Azure, http://www.windowsazure.com/

[3] KT Ucloud, http://ucloudbiz.olleh.com/

[4] Google Cloud Engine, https://cloud.google.com/

[5] NHN Entertainment, http://cloud.toast.com/

[6] OpenStack, https://www.openstack.org/

[7] Chavan, Vinay, and Parag Ravikant Kaveri. "Clustered virtual machines for higher availability of resources with improved scalability in cloud computing." Networks & Soft Computing (ICNSC), 2014 First International Conference on. IEEE, 2014.

[8] Whaiduzzaman, Md et al. "Cloud service selection using multicriteria decision analysis." The Scientific World Journal 2014 (2014).

[9] Wu, Qingtao et al. "A cloud service resource classification strategy based on feature similarity." Journal of Networks 9.11 (2014): 2987-2993.

[10] Ahn, Younsun, and Yoonhee Kim., "Semantic Cloud Resource Recommendation Using Cluster Analysis in Hybrid Cloud Computing Environment." KIPS research paper Computer and communication system volume 4, No. 9 (2015): 9.

[11] Moscato, Francesco et al. "An analysis of mosaic ontology for cloud resources annotation." Computer Science and Information Systems (FedCSIS), 2011 Federated Conference on. IEEE, 2011.

[12] SOM(Self-Organizing Map), Kohonen, Teuvo, and Timo Honkela. "Kohonen network." Scholarpedia 2.1 (2007): 1568.

[13] k-means algorithm, MacQueen, James. "Some methods for classification and analysis of multivariate observations." Proceedings of the fifth Berkeley symposium on mathematical statistics and probability. Vol. 1. No. 14. 1967.

[14] Elbowmethod,https://en.wikipedia.org/wiki/Elbow_method_(clustering)

[15] Computational fluid dynamics(CFD), Seoul National University ASDL(Aerodynamic Simulation Design Lab)

[16] Montage GALFA, http://montage.ipac.caltech.edu/docs/cubemosaicstutorial.html