클라우드 시스템의 지능적인 자원관리를 위한 적응형 부하균형 기반 그룹화 기법☆

Grouping Method based on Adaptive Load Balancing for the Intelligent Resource Management of a Cloud System

마테오 로미오* 양 현 호** 이 재 완***
Romeo Mark A. Mateo Hyunho Yang Jaewan Lee

요 약

클라우드 시스템에 대한 현재의 연구들은 대규모 시스템 구현에 있어서 클라우드 구성요소들 간의 적절한 상호작용에 집중되어 있다. 그러나 이러한 시스템들은 속성을 기반으로 한 유사한 서비스 제공자들을 그룹화 하거나 효율적인 자원공유를 향상시키기 위한 지능적인 부하분산과 같은 지능적 기법을 제공하지 않는다. 본 논문은 클라우드 제공자를 그룹화하여 효율적인 서비스 가상화를 제공하여 서비스 프로비저닝을 향상시킨다. 클러스터 분석에 기반한 클라우드 서비스 제공자의 그룹화는 유사하거나 관련된 서비스를 하나의 그룹으로 만든다. 동적인 부하 균형화는 클라우드 시스템의 서비스 프로비저닝을 지원하며 동적인 기법을 사용하여 그룹내에서 부하분산을 담당한다. 제안한 가상화 기법(GRALB)은 다른 기법에 비해 메시지 오버헤드나 성능 면에서 좋은 결과를 보였다.

ABSTRACT

Current researches in the Cloud focus on the appropriate interactions of cloud components in a large-scale system implementation. However, the current designs do not include intelligent methods like grouping the similar service providers based on their properties and integrating adaptive schemes for load distribution which can promote effective sharing of resource. This paper proposes an efficient virtualization of services by grouping the cloud providers to improve the service provisioning. The grouping of cloud service providers based on a cluster analysis collects the similar and related services in one group. The adaptive load balancing supports the service provisioning of the cloud system where it manages the load distribution within the group using an adaptive scheme. The proposed virtualization mechanism (GRALB) showed good results in minimizing message overhead and throughput performance compared to other methods.

☞ keyword : Cloud computing(클라우드 컴퓨팅), intelligent framework(지능형 프레임 워크), cluster analysis(클러스터 분석), adaptive load balancing(적응형 부하 균형)

1. Introduction

With the advances of the Internet, the perceived vision of computing to be widely used as pay-per-use like utilizing water, electricity, gas, and telephony is realized and the developments are still growing. This utility computing, which is also known as Cloud computing, provides a basic level of computing service that is considered to meet the everyday needs of the general community. Cloud infrastructures are popular in world wide to hub the

* 준 회 원 : 군산대학교 전자정보공학부 박사과정
rmmateo@kunsan.ac.kr
** 정 회 원 : 군산대학교 정보통신공학과 교수
hhyang@kunsan.ac.kr
*** 종신회원 : 군산대학교 정보통신공학과 교수
jwlee@kunsan.ac.kr (교신저자)
☆ This paper is supported by Industry and University Research Consortium of Small and Medium Business Administration (SMBA) of 2010
[2011/02/10 투고 - 2011/02/22 심사 - 2011/05/04 심사완료]

business applications and support open collaborations. In USA, a cloud computing, developed at NASA Ames Research Center, was built for public access[1]. The Nebula Cloud Computing Platform[1] is currently used for education and public outreach, for collaboration and also to support NASA's missions. In Europe, the European Union FP7 (Seventh Framework Programme) funded a cloud project which was the Reservoir[2] (Resources and Services Virtualization without Barriers). The Reservoir project enables a massive scale deployment and management of complex IT services across different administrative domains, IT platforms and geographies. Resources and services are transparently provisioned in the Cloud using virtualization technologies. However, these current designs did not include the method in finding similar cloud service providers to share resources and collaborate. Extracting information from the Cloud environment is helpful in the information-awareness of a cloud user or service provider to filter or categorize the services. Some researches studythe performance of the virtualization[3,4]. The profile parameters from virtualization can be used to optimize the load performance of a cloud infrastructure.

This paper proposes a virtualization mechanism which includes intelligent grouping of cloud service providers and providing an adaptive scheme for the load balancing technique of the group to support scalability and efficient management of resources in the Cloud. A cloud system is presented which is designed 1) to provide an effective service provisioning using agents in the cloud environment, 2) to provide a group method for cloud service providers to collect the similar or related cloud services and 3) to manage the load distribution of the cloud servers by using an adaptive scheme. This paper combines the grouping with the adaptive load balancing (GRALB) as the virtualization mechanism for the cloud system. The performance of GRALB is measured in message overhead and throughput performance compared to other methods.

2. Related Work

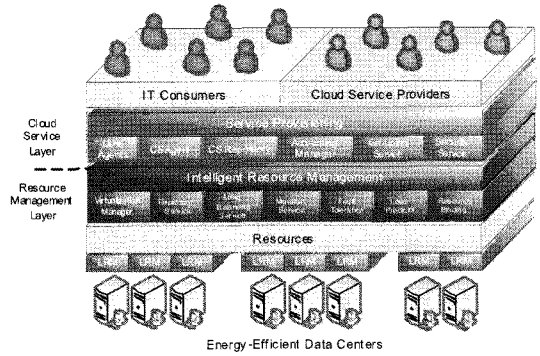
Implementing a group technique in a distributed system promotes scalable resources and collaboration from other resource providers. The architecture of a distributed object system in[5] supports the grouping of distributed objects for distributed applications. The appropriate algorithm for grouping is also considered in implementing a dynamic reconfiguration for distributed application. Serrano et al.[6] used clustering in object-oriented metrics techniques for the assessment of relationships and interactions between object-oriented constructs such as classes, objects, and methods. The clustering technique in[6] is used to create a hierarchical group that is convenient for guiding the allocation of the subsystems to a hierarchical network. To extend traditional fragmentation results of object oriented database systems, a methodology for the distribution design of object-based information systems is proposed in the research of Leclercq et al.[7]. In grid computing, the resources of many computers in a network are utilized efficiently to work on asingle task, usually to solve a scientific or technical problem that requires a great number of computer processing cycles or access to large amounts of data. Grouping of services is done to optimize the workflows of a Grid system in[8]. Moreover, the scalability of resources is studied in[9] where a semantic overlay network of service repositories exploits the semantics and group similar information. This allows repositories to be in different semantic groups thereby forming a group of coordinating

repository nodes. In[10] a scalable virtual organization (SVO) is proposed to support scalable resource sharing in virtual organizations (VO). The SVO tackles the scalability of resource sharing in MDS using a grouping technique and includes an efficient distribution of loads in GRAM. In the process of SVO, the proposed group similarity function (GSF) is used to determine similar VOs to perform merging of service information and requesting of additional resources. However, the Cloud environment is more diverse and more complex than Grid environment. The grouping technique should analyze accurately the contents of each cloud service and perform a complex procedure to acquire the proper grouping. It is assumed that grouping the correct cloud providers will implement the effective collaboration.

3. Cloud based on the Intelligent Resource Management

The proposed cloud architecture supports intelligent methods for service provisioning and resource management. A provisioned service is one that requires association of a user account and/or other information with the service [11]. In this paper, service provisioning is the automation of functions to acquire and deploy cloud service like initializing, metering the use, finding, grouping, etc. The layered structure of the proposed cloud architecture in the point of service provisioning and resource management is shown in Figure 1. Also, the proposed cloud architecture supports the brokering for cloud services and efficient virtualization of resources.

In the cloud service layer, a cloud service can be composed of several cloud services. The service



(Fig. 1) Layered approach of the proposed cloud architecture based on intelligent service provisioning and resource management.

providers can be independent in providing the resources for their cloud services and they can also rent resources to other resource providers or cloud servers. After a user agent finds the appropriate resource or service, the security service will process the authentication process. This process is also done in the grouping of cloud service providers. The resource management layer handles the virtualization of cloud services and assigning cloud services to be hosted on cloud servers. The layer is consisted of components that manage the cloud service resource allocation to the cloud servers. The local resource manager (LRM) manages the local hardware of a cloud server. An independent group of cloud serversprovides a small amount of storage and the dynamic resource sharing implements resource binding from independent resource providers. The following describes the components of proposed architecture.

3.1 Cloud Service Layer

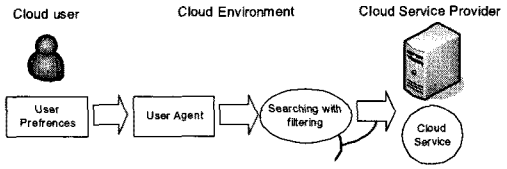
- User agent - used by a user to search for cloud services. It interacts with CS agents to process cloud services.
- Cloud service (CS) agents - cloud service

providers use this agent to interact with users, other cloud service providers, accounting manager, grouping service and virtualization manager.

- Cloud service (CS) repository - stores the information of cloud services. Cloud services are registered to this repository which is accessed by user agent.
- Accounting manager - handles the accounting procedures of using services and resources.
- Grouping service - groups the cloud service providers according to their cloud service properties. The CS agents invoke the grouping service to find the appropriate group.
- Security service - processes the authentication and authorization in accessing a resource.

3.2 Resource Management Layer

- Virtualization manager (VM) - handles the interaction between users, cloud service providers and resources. This is the main component in implementing the intelligent methods in the cloud environment.
- Replication service - handles the replication of the cloud service based on the service demand trendswhich is analyzed by the VM.
- Load balancing service - analyzes the load of the resources in a group. Also, the load balancing service checks the loads of LRM.
- Migration service - migrates cloud services based on the load analysis of the load balancing service.
- Fault tolerance - handles the disconnections and faults in the resource management layer.
- Load predictor - predicts the loads of the system to trigger the process of resource binding.
- Resource binding - provides a method of binding resources for the scalable resource sharing. The cloud service provider interacts with the resource



(Fig. 2) The interaction of cloud user in searching the appropriate cloud service using the preferences to filter.

binding after it joins a group.

- Local resource manager (LRM) - handles the local resource management of a single cloud server.

4. Grouping Cloud Providers with Adaptive Load Balancing

A user agent is used to interact with cloud service providers (CSP). The proposed cloud system integrates an intelligent algorithm to process the profiles of users in selecting the best cloud service for a request. Figure 2 shows the interaction of user agent to find the cloud services.

The selection of cloud services is processed in filtering using the user preferences. An input from the search method is represented by *in* and user preferences represented by *u* and *in* iscompared to the cloud service tags represented by t. An outcome value can be represented by a string or a numerical value. The process matches the cloud user preferencesand cloud service tags to process the filtering.

$$selected\_cloudservices = \sum_{n=1}^{N} f(u_n, t_n) \quad (1)$$

$$rx = \frac{t_{\max} - t_{\min}}{m} \quad (2)$$

$$f(u,t) = \{x_{\min} <= t < x_{\max} \mid x_{\max} = t_{\min} + (rx \times cat(u)), x_{\min} = x_{\max} - rx\}, \quad (3)$$
$$if\, cat(u) = m \quad then \quad x_{\max} = t_{\max} \,\&\, t <= x_{\max}$$

Equation 1 showsthe filtering function where all cloud services in N are the selected cloud services using the inputs (in) from the search. The u and t are compared using the function $f(u, t)$. The output values from u are categorical values and these are transformed in real numbers using Equation 2 and Equation 3. In Equation 2, the rx represents the length of each category from user preference which is used in transforming categorical values. The maximum value from N represented by t_{max} is subtracted to the minimum value represented by t_{min} and the result is divided by the number of categories represented by m. In Equation 3, the filtering function chooses all cloud services that satisfies the condition x_{min} <= t < x_{max} where x_{min} is the starting range value of u and x_{max} is the ending range value of u. The $cat(u)$ is a function that determines the index category of user preference. If m=3 then the indexes are $cat(u)$={1,2,3}. If the u is the last indexed category from the selection then the x_{max} will be equal to t_{max} and t will be filtered using the condition x_{min} <= t < x_{max}. The results from the filtering are listed in *selected_cloudservices* and returned to the user who will decide which service is appropriate for the search. The chosen cloud service means it satisfies the user's needs based on the inputs and user preferences.

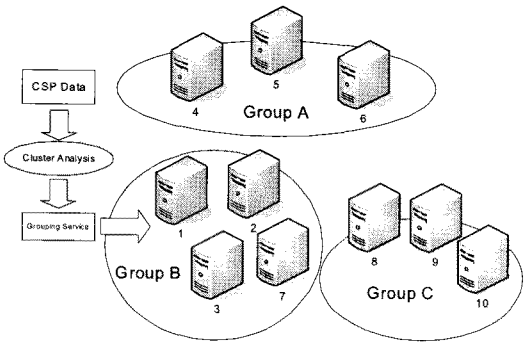
4.1 Grouping the Cloud Service Providers

The proposed cloud system supports the resource binding and groups the cloud service providers to provide additional resources and to prevent late responsesfrom high frequent accessed cloud services. Grouping is processed in cluster analysis where it groups CSP with similar property values. This approach assumes that the grouped CSPswill have a fast response onserving the large number of requests by sharing similar cloud services. The cloud service tags used in search of cloud services are also used for grouping. Every cloud service has tag values and the count of tags is used asproperty value in clustering. Each cloud service tag is incremented to summarize the CSP property and these are processed in the cluster analysis.

$$\min \sum J = \sum_{i=1}^{c} \left(\sum_{k=1, \mathbf{u}_k \in C_i} m_{ik} \left\| \mathbf{u}_k - \mathbf{c}v_i \right\| \right) \quad (4)$$

All property values (k) of a CSP are summarized and this is represented by u_k in the Equation 4. Optimizing the objective function in Equation 4 depends on the distance between vectors u_k and cluster centers cv_i where u_k is data and cv is the center value. m_{ik} represents the membership of the u_k which is {0,1} and C_i represents the cluster index. Equation 4is the objective function within cluster i where i is the index of the cluster. The J_i is minimized by several iterations and stopsif either theimprovement over the previous iteration is below a certain tolerance or J_i is below a certain threshold value. The algorithm to minimized J_i is composed of the following steps:

1. Place c points into the space represented by the objects that are being clustered. These points represent initial group center values (cv).
2. Assign each object (u_k) to the group that has the closest center value.
3. When all objects have been assigned, recalculate the positions of the c center value.
4. Repeat Steps 2 and 3 until the center values



(Fig. 3) Formation of virtual groups after the cluster analysis.

no longer move. This produces a separation of the objects into groups.

The center value will adjust every time there is a change in the members of the group and the calculation continues to minimize the J_i. Each cloud service provider will be assigned to a specific group based on the result of the cluster analysis. This processes the Euclidean distance on each group center value and the smallest value is the chosen group. After the grouping, the cloud service providers form a virtual group is illustrated in Figure 3.

$$selected\_group = \min\{\sum_{i=1}^{c} gsp_i(\mathbf{u})\} \quad (5)$$

$$gsp_i(u) = \sum_{j=1}^{J} \|u_j - cv_j\| \quad (6)$$

Whenever a cloud service provider wants to join a group to merge resources, it requests the grouping service to process its data and compare to other registered cloud service provider groups. This process enables a cloud service provider to determine the most appropriate group to merge its resources. Again, the cloud service tags are summarized by a vector value u to process the classification using the classification structure built from clustering in Equation 4.

In Equation 5, the u which is a summarized data of a cloud service provider that wants to join and this is processed in all groups in Equation 5 and each property from a group, $u = \{u_1, u_2,...u_j\}$, is calculated its Euclidean distance in Equation 6. The gsp_i is a group of cloud service providers that processes each u to be compared to its cv_j. The smallest value among the groups will be the selected group to join and merge the resources of a cloud service provider.

4.2 Adaptive Load Balancing

Loads are defined as queued tasks in a node and these are used by the load balancing service. The related and similar cloud services are collected in the group of cloud service providers using the group method. These grouped cloud services can serve replicas where all grouped cloud services are assumed to have the same functionality. In this paper, an adaptive approach by switching the two well-known methods in load balancing based on a load threshold value is used. At start, round-robin is activated by the local resource manager of a cloud server to access cloud services alternately within the cloud servers. Switching from round-robin to least load selection is determined if the load distribution status of all nodes presented by σ is greater than the load variation threshold presented by ΦL, otherwise, switch back to round-robin.

$$\mu_{pt} = \frac{1}{N}\sum_{i=0}^{N}\sum_{j=0}^{M} pt_{ij}, \quad (7)$$

$$\Phi_L = \frac{\mu_{pt}}{2} \quad (8)$$

Equations 7 and 8 show the calculation of the threshold value for load variation where *pt* is the processing time of a service *j* in a node *i*. In Equation 7, *M* is the total number of cloud service in a node, N is the total node in a group. In Equation 8, the threshold value is calculated by the mean of all processing time of services from nodes and divided by 2. This also means that more than the half of the mean value of processing time will trigger the execution of the least load selection scheme. Least load selection is activated when all nodes have load difference greater than ΦL. The load variation threshold value in Equation 8 is compared to the current load variance in Equation 9 to switch the load distribution scheme.

$$\sigma = \frac{1}{N}\sum_{i=1}^{N}(l_i - \mu) \qquad (9)$$

Equation 9 is the load variation which is based on the standard deviation of total current loads and is compared to the load variation threshold. l_i is the current load of a cloud server while the μ is the mean load value of all cloud servers. If another service was added in a node, then the load variation threshold also will be updated. An increase on Equation 9 indicates an increase on load variations among the cloud servers while performing the current load distribution scheme which will allow a selection of the least load distribution. A zero value from σ means there is equal distribution from all nodes which is almost impossible to acquire.

5. Experimental Evaluation

The proposed grouping with adaptive load balancing (GRALB) was integrated to the cloud system. The responsiveness of a cloud system using the proposed method is tested by measuring the system throughput performance. We used the traditional approaches in distributed system to compare the performance of service request forwarding. The round robin (RR), least load selection (LL), adaptive load distribution, round robin with grouping (RRG) and least load selection with grouping (LLG) were compared to the proposed method (GRALB) in message overhead and throughput performances. Throughout the simulation, the total message overhead was gathered to determine the amount of latency from exchanging messages and the total delay time from throughputs was gathered to determine the responsiveness of the system implementing the schemes. To set the configuration in measuring the throughput and network latency performance, the simVO [12] was used.

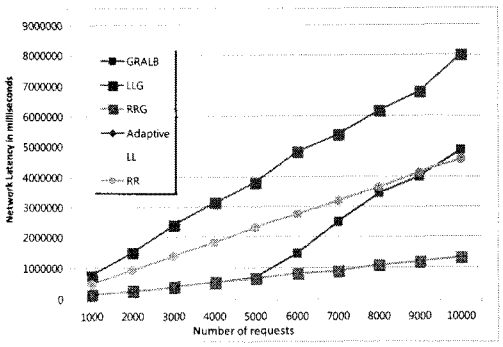
The simulation environment was consisted of 100 nodes, 20 services and 20 virtual groups. The nodes were divided equally into10 domains and identified by {A, B,⋯, J}. A ring topology was used to connect the domains where each domain was attached with two domains represented by {A↔B↔C↔D↔E↔F↔G↔H↔I↔J↔A} and each link has a latency of 10 milliseconds (ms). All nodes in a domain were connected to a router and their connections have a mean latency of 5 ms. Separated by domains, a node can connect to another node on a different domain by joining on avirtual group. Services were replicated throughout the nodes where each node was deployed with different 5 services. The assignment of nodes in a group and deployment of services in a node were random. Every group was set with initial time and termination time where each has 10,000 ms life time to execute and has a time interval of 500 ms to start after the other. When a virtual group is started in the simulation,

the calculation of cluster analysis is processed. If there are 3 current groups including the newly started virtual group then all nodes will be processed to cluster analysis. Virtual groups were labeled in the simulation, which were office workers, business collaborators, and poster/web designers and repeatedly used these labels to all 20 groups.

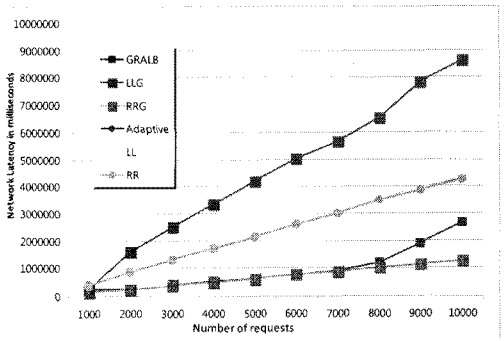
5.1 Simulation Result

The simulation was done in two cases. The first case generated the requests using a normal distribution of arrival time. This determines the performance of the algorithms in handling the request arrival in exponential increasing and decreasing manner. From the start of the simulation, few requests arrive until in the middle of the simulation period. After reaching the peak, the request arrivals decrease exponentially. The second case generated the requests using an exponential distribution of arrival time. This determines the performance of the algorithms to handle the request arrivals in an exponential increasing manner. In this case, the amount of requests in the end of simulation period is the highest. The message overhead and throughput performances used a volume of requests from 1,000 to 10,000 for evaluation and the results are shown in Figures 4, 5 and 6. In all figures, two cases are presented in the left and right sides. In the left is the normal distribution and in the right is the exponential distribution.

In Figure 4, the message overhead performances of all algorithms in two cases are shown. It was observed that the RRG has produced the least message overhead and the GRALB is secondin both cases. The reason why RRG has the least message overhead in both figures is because it does not have



(A)

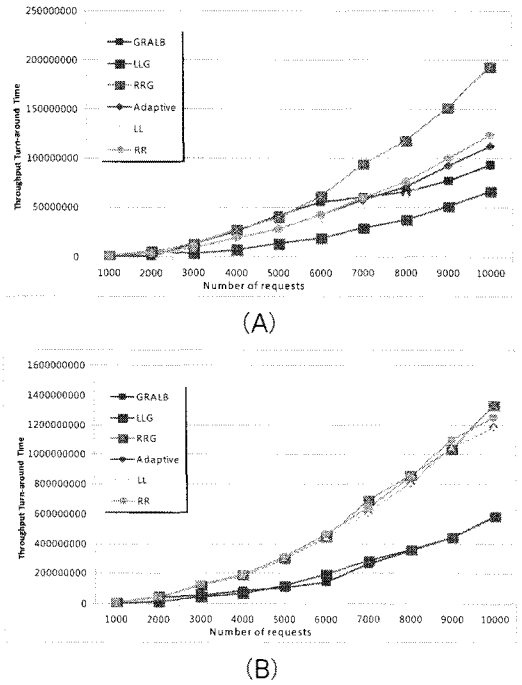


(B)

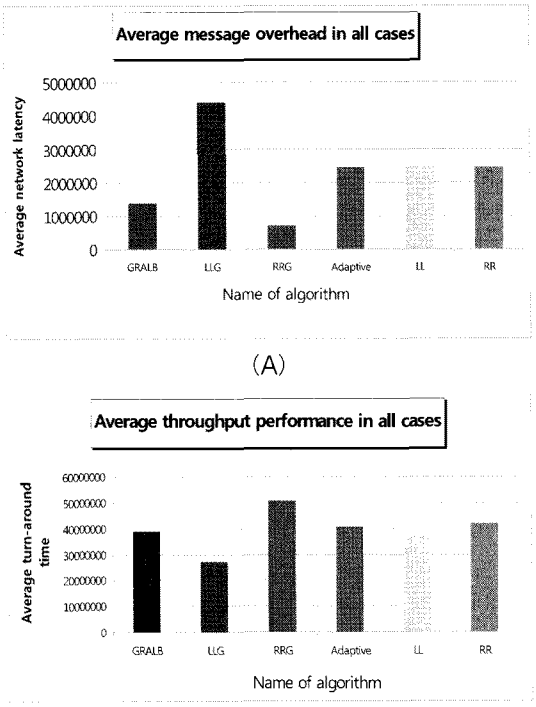
(Fig. 4) Message overheads by network latencies using the generated requests based on arrival time in normal distribution (A) and in exponential distribution (B).

an additional latency to decide which node it will distribute the loads which was the case in LLG. Also, it was observed from the GRALB, there was an increase in latency starting from 5,000 (Figure 4A) and 8,000 (Figure 4B) requests. In a large number of requests, the system frequently switches to least load selection because most of the time the current load variance is greater than the loadvariance threshold value.

In Figure 5B, the throughput performance shows that the GRALB is the fastest in the exponential case. The LLG has an opposite result in Figure 5 compared to Figure 4 which shows better in load distribution but still has the highest message



(Fig. 5) Total turn-around time of throughputs using the arrival time in normal distribution (A) and in exponential distribution (B).



(Fig. 6) Average message overheads (A) and throughput performance (B) from all cases

overheads. The RRG was inefficient in handling the response time of the request because it does not handle the high load variance which causes delays in the response time of the request. The average results of each algorithm from all cases are determined in Figure 6 which shows a summarized performance of all algorithms.

In Figure 6A, it is observed that RRG has the lowest message overhead and GRALB is second to RRG while the LLG has the highest message overhead. However, in Figure 6B, the RRG has an opposite result and is theslowest to response. The LLG is optimal in handling the task distribution, however, it has the worst message overhead shown in Figure 6A.

The performance of GRALB was compared using the ratio toother algorithm (e.g. LLG/SAG) where a value less than 1.0 means the algorithm compared has better performance than the DRALB. The DRALB was better than LLG (3.2), adaptive scheme without grouping (1.8), LL (1.8) and RR (1.8) in message overhead performance. RRG (0.7) produced the lowest message overhead but was the slowest in throughput performance. The DRALB was better than RRG (1.3), adaptive scheme without grouping (1.0), and RR (1.0) in throughput performance. LLG (0.7) and LL (0.9) has better response time compared to DRALB but produced high message overheads.

6. Conclusions

This paper presented a cloud framework based on the intelligent resource management which is composed of sub-procedures of intelligent techniques in managing resources. The intelligent methods were integrated in several components of the cloud service and resource management layers. This paper focused on the proposed virtualization mechanism which includes an intelligent grouping of cloud service providers and an adaptive scheme for the load balancing technique to support scalability and efficient management of resources in the Cloud. The grouping method was used to group the appropriate cloud service providers and the adaptive load balancing was used to distribute the loads in the group of cloud service providers. In the cloud service layer, the grouping method based on cluster analysis grouped the similar cloud service providers for efficient collaboration. The resource management layer executed the adaptive load distribution to handle the efficient task allocation.

The result showed that the proposed algorithm, which is the GRALB, performed better in message overhead than LLG, adaptive scheme, LL and RR. RRG had the lowest message overhead but the slowest in response time. Also, the GRALB performed better in throughput compared to than RRG, adaptive scheme without grouping and RR. LLG was the fastest to response on requests but had the highest message overhead.

References

[1] Project Nebula, http://nebula.nasa.gov

[2] Reservoir Project, http://www.reservoir-fp7.eu

[3] Casazza, J.P., Greenfield, M., and Shi, K.: Redefining Server Performance Characterization for Virtualization Benchmarking, Intel Technology, Vol. 10, No. 3 (2006).

[4] Iyer, R., Illikkal, R., Tickoo, O., Zhao, L., Apparao, P. and Newell, D.: VM3: Measuring, Modeling and Managing VM Shared Resources, Computer Networks, Vol. 53, No. 17 (2009) pp 2873-2887.

[5] Shin, C.S., Chung, Y.J. and Joo, S.C.: Distributed Object Group Framework with Dynamic Reconfigurability of Distributed Services, LNCS, 3251 (2004) pp. 121-128.

[6] Serrano, M.A., Carver, D.L. and Montes de Oca, C.: Mapping Object-Oriented Systems to Distributed Systems using Data Mining Techniques, LNCS, Vol.1821 (2000) pp. 79-84.

[7] Leclercq, E., Savonnet, M., Terrasse, M.N. and Yétongnon, K.: Object Clustering Methods and a Query Decomposition Strategy for Distributed Object-based Information Systems, LNCS, Vol. 1677 (1999) p. 817.

[8] Glatard, T., Montagnat, J., Emsellem, D. and Lingrand, D.: A Service-Oriented Architecture Enabling Dynamic Service Grouping for Optimizing Distributed Workflow Execution, FGCS Vol. 24, No. 7 (2008) pp. 720-730.

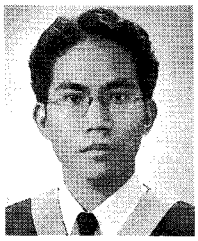
[9] Sapkota, B., Nazir, S., Vitvar, T., Tomay, I., Vasiliu, L. and Hauswirth, M.: Semantic Overlay for Scalable Service Discovery, International Conference on Collaborative Computing: Networking, Applications and Worksharing (2007) pp. 387-391.

[10] Mateo, R.M., Lee, J. and Lee, M.: Scalable Resource Sharing using Group Similarity Function in Grid System, Journal of Korean Society for Internet Information, Vol. 11, No. 4, (2010) pp. 73-83.

[11] Web Services Provisioning, http://www.ibm.com/developerworks/library/ws-wsht/?n-ws-1102

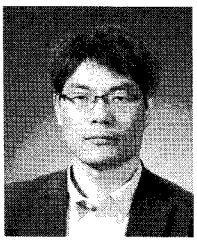
[12] simVO: A Simulator for Virtual Organization,
http://dslab.kunsan.ac.kr/en/simvo/
tmskim@korea.ac.kr

◐ 저 자 소 개 ◑



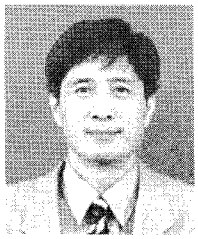
Romeo Mark Mateo

2004년 West Visayas State University, Philippines BS in Information Technology
2007년 Kunsan National University, South Korea, Master of Engineering major in Information and Telecommunications
2007~현재 Kunsan National University, SouthKorea, Graduate student in Ph.D course
관심분야 : Distributed systems, data mining, fuzzy systems, multi-agents, ubiquitous sensor networks, cloud computing
E-mail : rmmateo@kunsan.ac.kr



양 현 호 (Hyunho Yang)

1986년 광운대학교 전자공학과 졸업(학사)
1990년 광운대학교 대학원 전자공학과 졸업(석사)
2003년 광주과학기술원 정보통신공학과 졸업(박사)
1989년~1990년 삼성SDS 근무
1991년~1997년 포스데이타(주) 근무
1997년~2005년 순천청암대학 근무
2005년~현재 군산대학교 정보통신공학과 교수
관심분야 : 무선데이터통신, RFID/USN etc.
E-mail : hhyang@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, 실시간 시스템, 컴퓨터 네트워크, 클라우드 컴퓨팅 등
E-mail: jwlee@kunsan.ac.kr