ESBSC: Energy-aware Service Brokering Strategy in Clouds

Feng Ma
School of Computer and Information
Yunnan University of Finance and Economics
Kunming, 650221, China
E-mail: 375090618@qq.com

Xiaodong Liu
School of Computing
Edinburgh Napier University
Edinburgh, UK
Email: x.liu@napier.ac.uk

Ying Yang
International Business School
Yunnan University of Finance and Economics
Kunming, 650221, China
E-mail: 17789283@qq.com

Wei Yao, Jinjin Cai, Fang Wang
College of Information Science & Technology
Hebei Agricultural University
Baoding, 071000, China
Email: yaowei111@126.com

***Abstract*— One of the key features of cloud computing is on demand resource provision. Unfortunately, different cloud service providers often have different standards, which makes the job of choosing the suitable resource to be a very difficult one for the common users. Then the technology of service broker came out which is designed to choose the appropriate services among different providers to meet the user's requests. Besides get the final computing results, there are many other conditions often be taken into account, like energy consumption and response time. Maximizing the energy efficiency is not only good for the environment protection but also could benefit the user and provider financially. This paper proposed an energy-aware service brokering strategy, which aimed at finding a suitable trade-off between energy consumption and user satisfaction. The simulation results have shown that the proposed strategy can effectively reduce the level of energy consumption and meanwhile maintain the user satisfaction at a reasonably good level.**

Keywords—Energy-aware, Service Broker, Cloud Computing

I. INTRODUCTION

The cloud computing can provide almost unlimited on-demand resource for user's requests, and it often consists of many large Data Centres (DCs), which are located in different geographical regions. These DCs may belong to governments, enterprises, or other groups of organizations, and they are usually referred to as cloud service providers [1]. However, most cloud service providers have their own interface type, energy consumption level, cost policy and other service features [2]. It is very difficult for normal users to find out which one is the most suitable for their requests, therefore, the cloud service broker was proposed. According to Buyya [3], a broker is any service that acting as an intermediary mediating between users and providers of cloud services, offering its expertise in the evaluation of the proposals that are best suited to user needs and in the subsequent adoption or development of new products based on them.

Many researchers believe that using cloud service brokers to satisfy the user requirements is a promising research direction. Bossche proposed a broker scheduling method [4] which could perform operations in a hybrid cloud with feasibility and cost minimization. Wickremasinghe developed a tool called CloudAnalyst [5], to model and evaluate real-life problems deployed in the cloud, and at the same time, they also proposed two cloud service broker methods and three Virtual Machine(VM) load balancing algorithms on this tool for their own experiments.

With the fast development of cloud computing, more and more DCs had been established and inevitably consuming more and more energy. It has caused two main issues, firstly high carbon emissions which are demage the environment and cause financial loss to the cloud service providers. According to Qureshi [6], a 3% reduction in energy cost for a large company like Google can be transformed into over a million dollars in cost saving. In other words, making the cloud service broker more efficient is not only beneficial for the environment protection but also will benefit the cloud service providers. Therefore, this paper proposes a strategy which would make energy consumption more efficiently and meanwhile maintain the user's satisfaction. The strategy mainly contains two parts: the energy-aware brokering algorithm and the load balance algorithm. The first algorithm aims at allocating the user's request among different DCs and

This work is supported by National Nature Science Foundation under Grant 61461051.and YUNFE Project Grant YC2015D16.

maintaining a low level energy consumption. Whilst the later one intends to create a more reasonable load balance between VMs, and make the overall user satisfaction to a suitable level.

The rest of the paper is organized as follows: Section 2 presents the related work of this issue, Section 3 describes the proposed broker strategy, Section 4 details the simulation setup and discusses the results, and Section 5 presents the conclusions and future work.

II. RELATED WORK

With the increased popularity of s cloud computing, a large amount of energy has been consumed by the numerous devices in DCs every day. Making energy consumption more efficient could benefit both the environment and the cloud service providers [6]. Several authors noticed the importance of a proper adoption and analysis of energy consumption models to assess the paradigm shift, and try to cope with the energy issue from different perspectives. Some researchers believed that with the help of new developed technologies, such as lower-voltage equipment and integrated systems by green industries such as ARM or Tensilica Inc. [7-8], or the design of power-aware computing practices at both hardware and software levels, e.g. the Green Flash project at Berkeley [9], this problem could be solved properly. Others pay great attention to the improvement of the energy to efficiency ratio, by using advanced cooling technology and power generation equipment [10,11], or the method of mixing the electricity produced from dirty or dangerous sources with renewable sources, as Yahoo and Google are doing (Greenpeace). Recently, an increased number of researchers began to focus on the design of algorithms and methodologies for energy efficient job scheduling [12, 13]. That is mainly because, the cloud computing could provide elastic and scalable service based on virtualization technology. Thus, well designed algorithms or methodologies may significantly improve the energy efficiency. For example, Hadji and Zeghlache proposed a minimum cost maximum flow algorithm [14], which effectively managed the energy consumption by dynamic workloads and flow variations. Inspired by the queening theory, Lin proposed a custom workload prediction algorithm [15] to deal with this issue. There are also many other work had been done in this field [16-18].

One of the issues of improving energy efficiency in cloud computing is that it might conflict with the user satisfaction. As noted by other authors in [19], the actual challenge is trying to maintain a satisfactory level of performance without increasing the energy costs. User's satisfaction may consists of several parts, such as the response time of their request, cost for the service, refused rate and so on. The cloud service broker comes out along with the development of cloud computing, as it can help users to efficiently find out the resource they need without concerned about any of the specify interfaces provide by different cloud service providers. A lot of work has been done in this field. A cloud brokering architecture for VM management in a hybrid cloud computing environment was proposed by Tordssonetal [20], and a cloud coordinator architecture that works in private and public cloud environments was proposed by Calheiros [21]. Kessacietal proposed a cloud brokering algorithm call Pareto Multi-Objective Genetic Algorithm for Cloud Brokering (MOGACB) [22], which focused on the broker's revenue as well as the user's satisfaction. According to them, among the several factors, response time often took a large portion in the final user's satisfaction.

However, few of these aforementioned strategies simultaneously concern about both the energy efficiency and the user satisfaction. Hence, in this paper we focus on proposing an effective solution to this drawback. The main contribution of this paper can be concluded in two key points. Firstly, an energy-aware broker policy is proposed, which will significantly reduce the level of energy consumption. Secondly, a load balance method is developed as part of the strategy, which will effectively allocate the user requests among different VMs to accelerate the overall performance, so that the user satisfaction level can be maintained at a reasonably good level.

III. THE PROPOSED BROKERING STRATEGY

As widely known, cloud computing has often been divided into three service layers, SaaS, PaaS and IaaS. The SaaS (Software As a Service) usually deals with the requirements of application execution. The PaaS (Platform As a Service) acts as a middleware and establishes a bridge between the infrastructure and the application requirements. The IaaS (Infrastructure As a Service) is response for the infrastructure management, such as the number of processors, the size of memory, the configuration of virtual machines and network connectivity. Our proposed broker strategy is in the IaaS layer since it needs to concern about the availability of system resources and the energy consumption.

The energy-aware service brokering strategy integrally includes two parts, namely Energy-Aware Brokering Policy (EABP) and State Aware Load Balance Algorithm (SSLB).

A. Description of EABP

The main job for the broker is to find out the suitable resource to the user requests, and different users often have different constraints with their requests, such as cost, energy consumption, response time and so on. As mentioned before, the energy consumption has become more and more important in the environment protection and the overall energy saving is also beneficial for the service providers. The algorithm proposed in this part mainly concerns the overall level of energy consumption.

The general process of a cloud service broker is that, it would put all of the user requests in a queue and then select the proper policy to make the next move as show in Figure 1. Since different DCs often have different energy consumption levels, the greediest policy may seem to put all the requests to the DC which has the lowest energy consumption. However, if we take a second thought and measure the energy consumption from the overall perspective, it might not be the best policy. That is because it may let a large amount of requests waiting in the queue, and make rest of the DCs in idle state and continuously consume energy.

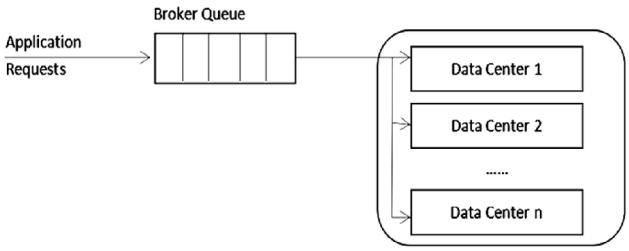

Fig 1. Broker process queue

The main idea of the first algorithm, known as Energy Aware Brokering Policy, EABP, attempts to properly allocate the user's requests to DCs and maintain an overall low level of energy consumption. To make it easy to follow, here we would like to use some symbols in the description of the algorithm. An object is defined as ObjDc to describe the features of a DC. It contains several important elements such as the resources, energy consumption level and name of DC, which are respectively noted as RE_{dc}, E_{dc} and N_{dc}. As we know, due to the virtualization technology, a DC could process more than one request at the same time, but this does not mean a DC could deal with an unlimited number of requests at the same time. In fact, every DC has its own threshold, and when the number of requests exceeds this limit, the forthcoming request may need more time to be processed or even be rejected by the DC as it does not have enough resources to deal with so many requests. Therefore, we use R_{dc}, T_{dc} and S_{dc} to present the number of requests, the threshold and the state of the DC. If number of requests has reached the threshold, S_{dc} will be set as B (BUSY), which means it should not take any more requests. Otherwise, we set S_{dc} as A (AVAILABLE).

The process of EABP could be described as follows: first it takes one user request from the broker queen, noted as RQ_i, where i is the index of the selected request in the queen; and let's use $reRQ_i$ to represents the resource it requires. Then, it initialise an object arrays of the DC, noted as $O_{dc}[n]$, where n stands for the number of DCs, and sort the array by S_{dc}. In the next step, we need to find out the lowest E_{dc} among those DCs which S_{dc} is A, and get its DC's name, noted as dcName. Then, the algorithm adjusts the R_{dc} of selected DC and rechecks all DCs S_{dc}. If anyone's R_{dc} reaches T_{dc}, change the value of R_{dc} to B, otherwise sets it as A. Finally, the algorithm repeats the above steps until all the requests in the broker queen have been processed. The following part is the description of EABP in pseudocode.

Algorithm 3.1 EABP
Input: RQ_i, $O_{dc}[n]$
Output: dcName
Begin
For (j=0;j<n;j++)//j represents the index of DC
Inital $O_{dc}[j]$;//set up its elements value
Sort O_{dc} by S_{dc} ;
temp = 0;
For (j=0;j<n;j++)
If ($reRQ_i$ <= $O_{dc}[j].RE_{dc}$ && $O_{dc}[j].E_{dc}$ <$O_{dc}[temp].E_{dc}$ && $O_{dc}[j].S_{dc}$ ==A)
temp = j;
dcName = $O_{dc}[temp].N_{dc}$;
$O_{dc}[temp].R_{dc} = O_{dc}[temp].R_{dc} + 1$;
For (j=0;j<n;j++)
If K requests has been process since late time check in DC j
$O_{dc}[j].R_{dc} = O_{dc}[j].R_{dc} - k$
If ($O_{dc}[j].R_{dc} < O_{dc}[j].T_{dc}$)
$O_{dc}[j].S_{dc} = B$;
Else
$O_{dc}[j].S_{dc} = A$;
End

B. Description of SSLB

A well designed broker strategy often needs to take the user satisfaction into account. Beside the cost, the response time has often been seen as a critical factor. Here, the response time refers to the time starting from a user request being sent out and ending at when the request has been processed and sent back to the user. Technically, the response time often consists of three parts: the time for the data transferring through network, the waiting time in the queen, and the processing time in DC. The first part largely depends on the network latency, and we have made some effort and discussion on the second part in section 3.1. Therefore, here we would focus on how to reduce the third part to get a better overall response time.

As well know, the DC often has a large amount of physical machines and every physical machine would use virtualization technology to build lots of VMs. To make it work more efficiently, a load balancer is often used in the physical machine, and the architecture of the system is show in Figure 2. If the load balance algorithm could effectively assign the user's request to a suitable VM and keep all VMs working in the same level of load, it would be very benefit for reducing the overall DC processing time.

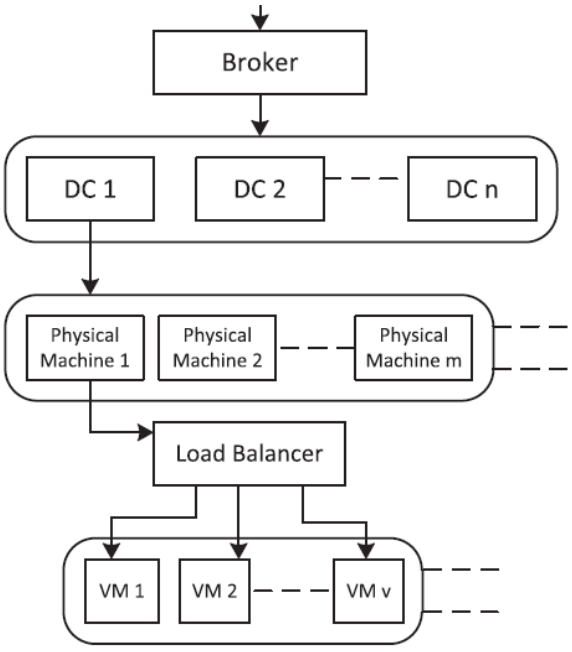

Fig 2. Architecture of the Load Balancer

The load balance algorithm, namely State Aware Load Balance Algorithm, (SSLB) proposed in this paper, attempts to achieve the goal by maintaining several lists to

record the state of the current VMs. Like the former algorithm, we build an object array to record the features of VMs, noted as $O_{vm}[m]$, where m is the number of VMs in the selected physical machine. R_{vm}, T_{vm} and S_{vm} respectively stands for the number of requests, the threshold value and the state of the current VM. A new factor is introduced to record the number of requests that a VM has processed and it is noted as RN_{vm}.

In SSLB, we first check all those S_{vm} is still A, if the R_{vm} could meet the $reRQ_i$, then find the VM which has the smallest RN_{vm}. Since a VM usually has a serial number instead of a name, we record the selected VM by send its serial number to vmSN. Thenext step is to update all of the VMs states which are under the same physical machine and wait for the next user request. The whole algorithm is described as follows in pseudo code.

Algorithm 3.2 SSLB
Input: RQ_i, $O_{vm}[m]$
Output: vmSN
Begin
For (j=0;j<m;j++)//j represents the index of VM
Inital $O_{vm}[j]$;//set up its elements value
temp = 0;
For (j=0;j<m;j++)
If ($reRQ_i$ <= $O_{vm}[j].RE_{vm}$ && $O_{dc}[j].RN_{vm}$ <$O_{dc}[temp].RN_{vm}$ && $O_{dc}[j].S_{dc}$ ==A)
temp = j;
vmSN = $O_{vm}[temp].SN_{vm}$;
$O_{vm}[temp].R_{dc} = O_{dc}[temp].R_{dc}$ +1;
For (j=0;j<m;j++)
{
If K requests has been process since late time check in VM_j
{
$O_{vm}[j].R_{vm} = O_{vm}[j].R_{vm}$ -k;
$O_{vm}[j].RN_{vm} = O_{vm}[j].RN_{vm}$ +k;
}
If ($O_{vm}[j].R_{vm} < O_{vm}[j].T_{vm}$)
$O_{vm}[j].S_{vm}$ = B;
Else
$O_{vm}[j].S_{vm}$ = A; }
End

IV. CASE STUDY AND DISCUSSION

A. Case Study

We used CloudSim [23] to construct our simulation environment. It is a classical tool in simulating the framework for cloud computing, and it supports large scale cloud simulation even in federated clouds. We use it to build the DCs and the VMs. For the convenience of later comparison, we built five different scenarios. The number of DCs in each scenario grows from 2 to 10, adding 2 DCs at a time. The physical hosts in each DC have the same configuration except for the energy consumption level. The detail of the physical host configuration is specified in Table 1.

Table 1. Physical Machine Configuration

Name	Value
CPU speed	10,000MIPS
CPU core	16
RAM	32GB
Storage	1000GB
Network bandwidth	10Gbps
CPU architecture	X64
OS	Linux
Virtual machine monitor	Xen
VM CPU speed	1000MIPS
RAM	1GB
Storage	50GB
Network bandwidth	100Mbps

For the generation of user requests, we used the Facebook user statistics throughout the world. From the statistics on 30 Oct 2016, Facebook had 1.79 billion monthly active users worldwide, and approximately 84.9% of daily active users are outside the US and Canada [24]. We assumed that each active user generates a new request in every 3 minutes. The bandwidth and latency of the network are generated using a Poisson distribution, and only the mean values are used. The same technique was engaged to produce the energy consumption level for each DC.

We used the Round Robin policy as the baseline to compare the strategy proposed by this paper. As our strategy has two parts, EABP and SSLB, we would like to introduce Round Robin policy into different parts to find out the different results. Therefore, the RR standing for the traditional Round Robin in both parts which represents a reference account of the energetic consumption level without the adoption of any energy aware mechanism. RR-SSLB notes for the strategy which use RR only in the former part, while EABP-RR notes for the strategy which use RR only in the later part. And the EABP-SSLB notes for the strategy proposed in this article.

As for the level of energy consumption, the simulation result is shown in Figure 3. In each scenario, the RR strategy's result is set at the baseline as 100%.

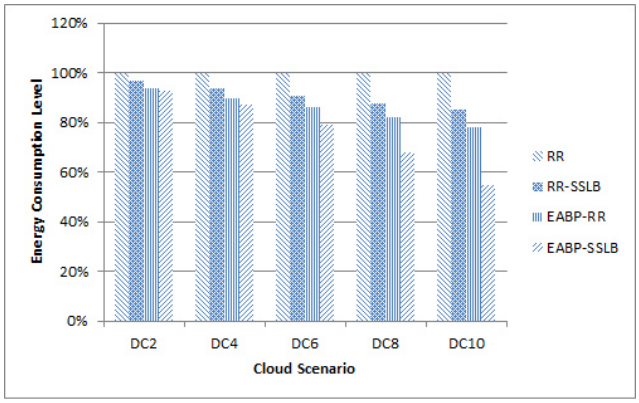

Fig 3. Energy Consumption Level

When coming to the response time, the average response time is introduced into the result as it is more suitable for the overall evaluation. The same scenarios were introduced into this example and the results are show in Figure 4.

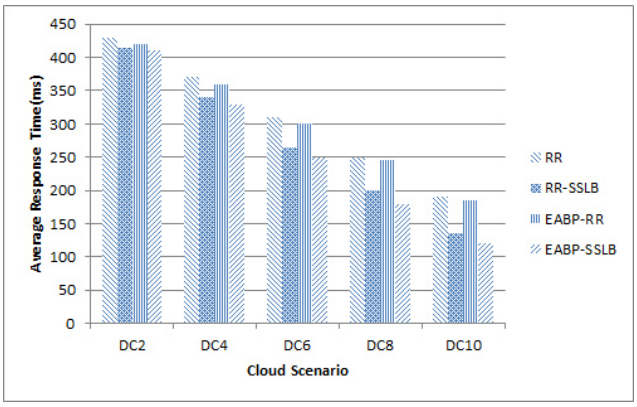

Fig 4. Average Response Time (ms)

Besides the response time, the processing time which just counts the time when requests are processed in DC, is another important criterion in this field. The results are shown in Fig 5.

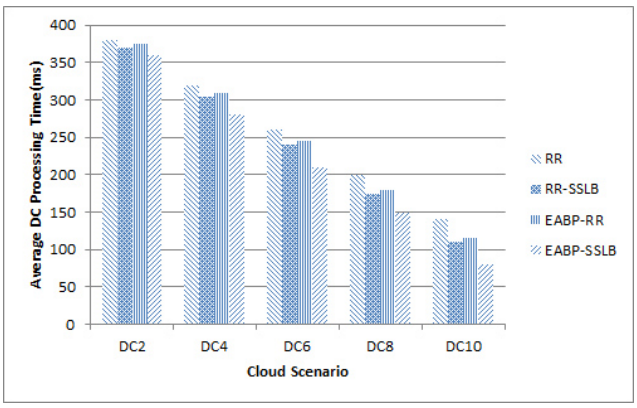

Fig 5. Average Processing Time (ms)

B. Discussion

From Figure 3, we can see that compared with RR, the EABP-SSLB strategy performs better, and the more DCs involved the more energy saved. That is because, as mentioned early, the energy consumption level of each DC is generated using a Poisson distribution, and more numbers means more chance for the proposed strategy to find out the low level energy consumption. RR-SSLB and EABP-RR are at the middle level which verifies that each step of the proposed policy is benefit for the overall energy consumption.

As shown in Figure 4, the EABP-SSLB strategy also gets a reasonably positive result in the average response time, but the between RR-SSLB and EABP-RR, the former has much better performance. After further analysis, we found that in some special scenarios, such as where there are few DC could meet the most request's requirement and they happen to be the low level energy consumption at the same time, EABP might concern too much about the energy consumption level and let lots of request gathered at several low level DCs especially when the number of DCs increases. Without the effort of SSLB, it might have the similar result with the traditional RR policy. However, the simulation results show that EABP-SSLB could manage to maintain the average response time at a reasonably good level.

V. Conclusion and Future Work

In this paper, an energy-aware service brokering strategy in cloud computing is proposed to manage the level of energy consumption and the user satisfaction at the same time in a balanced manner. The strategy consists of two algorithms, namely Energy Aware Brokering Policy (EABP) and State Aware Load Balance (SSLB) algorithm. By referring to the threshold and the state of every DC, EABP prioritizes the distribution of user requests to those with low level energy consumption. Therefore, the overall energy consumption could be reduced. SSLB is mainly concerned about the load balance between VMs inside a selected physical machine, as it could effectively influence the process time of every request and finally made a good effort to the whole response time.

Simulations results have shown that compared with the Round Robin policy, our proposed strategy has a better performance. However, if we take the strategy's two algorithms apart, the simulation results suggest that, in some special situation, EABP might lost its advantage. How to overcome these problems would be the future work.

References

[1] D. Bernstein, E. Ludvigson, K. Sankar, S. Diamond, M. Morrow, Blueprint for the intercloud-protocols and formats for cloud computing interoperability. *The Fourth International Conference on Internet and Web Applications and Services,* IEEE, Vol.13, No.2, 328-336, 2009.

[2] C. Vecchiola, X. Chu, M. Mattess, R. Buyya, Aneka—integration of private and public clouds, *Cloud Computing Principles and Paradigms*, Wiley, Hoboken, 251-274, 2011.

[3] Q. Buyya, J. Broberg, A.M.Goscinski (eds), *Cloud Computing: Principles and Paradigms*, Wiley Press, 2011.

[4] R. Van den Bossche, K. Vanmechelen, J. Broeckhove, Cost-optimal scheduling in hybrid IaaS clouds for deadline constrained workloads, *the 3rd IEEE International Conference on Cloud Computing*, 228-235, 2010.

[5] B. Wickremasinghe, R. N. Calheiros, R. Buyya, Cloudanalyst: a cloudsim-based visual modeler for analyzing cloud computing environments and applications, *the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, 446-452, 2010.

[6] G.A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, Cutting the electric bill for Internet-scale systems, *Proceedings of the ACM SIGCOMM2009 conference on Data communication*, 123-134, 2010.

[7] ARM Cortex-M Series, http://www.arm.com/products/processors/cortex-m/index.php

[8] Tensilica, http://www.tensilica.com/products/xtensa-customizable#overview

[9] D. Donofrio, L. Oliker, J. Shalf, M. F. Wehner, C. Rowen, J. Krueger, S. Kamil, M. Mohiyuddin, Energy-Efficient Computing for Extreme-Scale Science, *IEEE Computer*, Vol. 42, No.11, 62-71, 2009.

[10] IBM, 2010. Aquasar project. Available at: http://www-03.IBM.com/press/us/en/pressrelease/32049.wss

[11] SuperMUC, supercomputer hosted at Leibniz Supercomputing Centre. Description available at: http://www.lrz.de/services/compute/supermuc/systemdescription/

[12] S. Zikos, , H.D Karatza, Performance and energy aware cluster-level scheduling of compute-intensive jobs with unknown service times, *Simulation Modelling Practice and Theory*, Vol. 19, No.1, 239-250, 2011.

[13] A. Galizia, A. Quarati, Job allocation strategies for energy-aware and efficient Grid infrastructures, *The Journal of Systems and Software*, Elsevier, Vol. 85, No. 7, 1588-1606, 2012.

[14] M. Hadji and D.Zeghlache, Minimum cost maximum flow algorithm for dynamic resource allocation inclouds, *Proceedings of the IEEE 5th international conference on cloud computing*, 2012.

[15] M. Linetal, Dynamic right-sizing for power-proportional DCs. *Proceedings of the IEEE/ACM transactions on networking*, 1378-1391, 2013.

[16] S. Srikantaiah etal, Energy aware consolidation for cloud computing, *Cluster Computing*, Vol.12, No.1, 1-15, 2009.

[17] T.Knauth and C.Fetzer, Energy-aware scheduling for infrastructure clouds, *Proceedings of the IEEE 4th international conference on cloud computing technology and science*, 58-65, 2012.

[18] S. Xavier, J. Lovesum, A survey of various work flow scheduling algorithms in cloud environment, *International Journal Sci Res Pub*, Vol.3, No.2, 2013.

[19] I.S. Moreno, Jie Xu, Energy-efficiency in cloud computing environments: towards energy savings without Performance degradation, *International Journal of Cloud Applications and Computing (IJCAC)*, Vol.1, No.1, 17-33, 2011.

[20] R.N. Calheiros, A.N. Toosi, C. Vecchiola, R. Buyya, Acoordinator for scaling elastic applications across multiple clouds, *Future Generation Computer System*, Elsevier, Vol. 28, 1350-1362, 2012.

[21] J. Tordsson, R.S. Montero, R. Moreno-Vozmediano, I.M. Llorente, Cloud brokerng mechanisms for optimized multiple providers, *Future Generation Computer. System*, Vol. 28, 358-367, 2012.

[22] P.V. Krishna, Honey bee behavior inspired load balancing of tasks in cloud computing environments, *Appl. SoftComput*. No. 13,2292-2303, 2013.

[23] R.N. Calheiros, R. Ranjan, A. Beloglazov, C.A. DeRose, R. Buyya, CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, *Software:Practice and Experience*, Vol.41, 23-50, 2011.

[24] Facebook User Statistics 2016, http://newsroom.fb.com/company-info/, 2016.10.30.