JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 33, 1-16 (2017)
DOI: 10.6688/JISE.2017.33.1.1

Energy-Aware Resource Management in Cloud Computing Considering Load Balance

HEYANG XU AND BO YANG
School of Computer Science and Engineering
University of Electronic Science and Technology of China
Chengdu, Sichuan, 611731 P.R. China
E-mail: xuheyang124@126.com; yangbo@uestc.edu.cn

Energy-aware resource management in cloud computing has attracted lots of attention and many approaches have been proposed. A commonly used technique is server consolidation, which consolidates virtual machines (VMs) on a fewer physical servers and switches idle servers to low-power modes. Nevertheless, the majority of the proposed approaches do not consider load balance of active servers, which is an important issue that should not be ignored. In this paper, we investigate the problem of energy-aware resource management in cloud computing by taking load balance into account and formulate this problem as a multi-objective optimization model. The first optimization objective is to minimize the number of active servers and the second one is to balance the loads among these servers. Based on the proposed optimization model, a heuristic-based algorithm called *greedy-based load balance* (*GBLB*) *algorithm* is developed. Since reducing the number of active servers generally increases the number of VM migrations, we further minimize the number of VM migrations in the proposed GBLB algorithm. Simulation results show that, compared with other three popular algorithms, the proposed GBLB algorithm can reduce the number of active servers and achieve the best load balancing level at the cost of a few more migrations.

Keywords: resource management, cloud computing, load balance, virtualization, server consolidation, energy efficiency

1. INTRODUCTION

Cloud computing is a large-scale distributed computing paradigm supported by state-of-the-art data centers, in which a pool of computing resources is available to users (called cloud consumers) via the Internet [1]. In recent years, increasing demand for computational resources has led to a significant growth in the number of cloud computing servers, along with almost a double of the energy consumed by these servers and cooling infrastructures that support them [2]. Their share of power consumption is approximately between 1.1% and 1.5% of the total electricity used worldwide and is projected to rise even more [3]. High energy consumption gives rise to a large amount of operational cost which can accumulate more than the construction cost of servers and infrastructures in a short period [4]. This indicates the need for cloud service providers to adopt energy efficient resource management approach to ensure that their profit margin is not dramatically reduced due to high energy costs [5, 6].

Virtualization technology is widely adopted in cloud systems [7]. It can provide performance isolation between applications sharing the same resource and allow cloud providers to create multiple virtual machines (VMs) on a single physical server. By using

Received December 4, 2015; revised April 29, 2016; accepted June 28, 2016.
Communicated by Shiao-Li Tsao.

live migration [8, 9], VMs can be dynamically consolidated on a fewer physical servers and idle servers can be switched to low-power modes, *i.e*., sleep mode. This approach is known as server consolidation, which is seen as an efficient solution to increase resource utilization and reduce electric energy consumption in cloud systems [10].

Many efforts have been placed on the research of server consolidation and most of them focus on minimizing the number of physical servers. For example, Beloglazov *et al*. [5] investigate the problem of energy-aware resource allocation of cloud data centers and propose several heuristic algorithms to reduce the energy consumption via dynamic allocation of VMs. Ferreto *et al*. [10] formulate server consolidation problem as a linear programming model which tries to minimize the number of active servers and introduce several heuristic-based algorithms to control VM migration. Wolke and Pfeiffer [11] adopt several well-known vector bin-packing heuristics to address the issue of VM consolidation with the objective of minimizing the number of physical servers. However, load balance is also an important issue in cloud computing system [12]. Imbalanced load of the physical servers will cause more VM migrations which could result in extra energy consumption [13]. Therefore, an energy efficient consolidation approach cannot use eager migrations which minimize the number of physical servers, and ignore the issue of load balancing. In view of this, we consider the problem of energy-aware resource management taking into account load balance. A multi-objective optimization model is proposed with the first optimization objective of minimizing the number of active physical servers and the second one of minimizing the standard deviation of the CPU utilizations of the active servers. Based on the proposed multi-objective optimization model, we develop a heuristic-based algorithm called *greedy-based load balance* (*GBLB*) *algorithm*.

Server consolidation is achieved by VM migration, which is likely to cause negative effects on the service levels of application running on the migrated VM. Many previous studies [14-16] have evaluated this effect and concluded that the overhead of VM migration is small and can be acceptable in most cases. Also, migration costs may vary for different workloads due to the variety of VM configurations and workload characteristics [16]. Therefore, the overhead of VM migration is not formulated in the optimization model. However, the proposed GBLB algorithm tries to reduce this impact by adopting minimum VMs selection policy. We also perform extensive simulations with three synthetic workloads, *i.e*. low workload, medium workload and high workload, and a real world cloud trace to evaluate the performance of our algorithm.

The remainder of this paper is organized as follows. Section 2 reviews some significant related contributions in the literature. Section 3 introduces the system model used in this paper. Section 4 presents the multi-objective optimization model and introduces the details of the proposed algorithm. Simulations, experimental results and analyses are given in Section 5. We conclude our paper in Section 6.

2. RELATED WORK

In recent years, extensive efforts have been put into the research on server consolidation in cloud computing environments. Kimbrel *et al*. [17] investigate dynamic placement of web applications and introduce a heuristic algorithm, which tries to satisfy all the applications' demand while changing the applications' location as little as possible.

Karve *et al*. [18] further take into account balancing the resource utilization across all servers and introduce a heuristic algorithm which adds rebalancing placement to Kimbrel's algorithm. Tang *et al*. [19] adopt min-cost max-flow algorithm to optimize the relocation of applications. Tian *et al*. [20] extend the above researches, further minimize the resource utilization of servers in the worst case and introduce an approximation algorithm. These studies have investigated the dynamic application placement problem which is similar to the consolidation of VMs in cloud computing environments. In these studies, the energy efficiency issue of resources is not considered.

In order to address the energy efficiency issue, many energy-aware resource management approaches have been proposed. Lin *et al* [21] adopt dynamic voltage frequency scaling (DVFS) scheme and propose energy-aware task scheduling algorithms that can leverage per-core DVFS for multi-core platform. Kim *et al*. [22] investigate power-aware provisioning of VMs for real-life services and propose three DVFS-based RT-VM provisioning heuristics. Bobroff *et al*. [23] present a server consolidation algorithm, Measure-Forecast-Remap (MFR), to reduce the amount of required physical resources without violating VMs' performance. MFR algorithm predicates resource demand of VMs and dynamically remaps VMs to minimum number of active physical servers at each interval. These studies investigate the problem of energy-aware resource management approaches by only considering the CPU utilization without taking into account multiple resources, which are more suitable for real-life scenarios.

To further take into consideration multiple resources, Cao *et al*. [4] propose a power-saving approach based on demand forecast for allocation of VMs. The authors use a modified knapsack algorithm to find appropriate allocation between VMs and servers. Speitkamp and Bichler [24] introduce a linear programming model which minimizes server costs under a set of constraints for server consolidation problem and propose an LP-relaxation based heuristic. Sharifi *et al*. [25] propose four models to identify the performance interferences between processor and disk utilizations and the costs of migrating VMs. They develop an energy-aware scheduling algorithm using a set of objective functions. Perumal and Subbiah [26] propose a power-conservative server consolidation algorithm for resource management in cloud systems. The above research works have shown the effectiveness of the proposed algorithms in saving energy through extensive simulation-based experiments.

3. SYSTEM ARCHITECTURE

The active entities in a cloud computing system are users (resource consumers), resource providers and cloud scheduler [27, 28]. Users submit their jobs to the cloud scheduler. Resource providers can offer their resources to execute jobs submitted by users. The cloud scheduler is responsible for scheduling each job to an appropriate resource. Each user job is encapsulated into a VM with certain amount of CPU and memory requirements. Multiple VMs can concurrently run on a single physical server. The physical servers offered by resource providers are heterogeneous, which means that servers may have different memory capacity and CPU capacity, in terms of million instructions per second (MIPS). In a cloud computing system, physical servers communicate with each other via a fully inter-connected network such that there is a connection

between each pair of physical servers. By using live migration, VM can be dynamically migrated to any other physical server in the system in runtime.

Without loss of generality, suppose that a cloud computing system consists of m heterogeneous physical servers, denoted by $\mathbf{P}=\{p_1, p_2,\ldots, p_m\}$, and there are n VMs, denoted by $\mathbf{V}=\{v_1, v_2,\ldots, v_n\}$, running in the system. The jth physical server (denoted by p_j) has certain amount of CPU capacity and memory capacity, which can be obtained by its configuration information. Denote by Ω_j and Γ_j the CPU capacity and memory capacity of p_j, respectively. The CPU demand and memory demand of the ith VM (denoted by v_i) are represented by ω_i and γ_i, respectively. In this paper, we assume that the resource demands of VMs are constant during their runtimes and this assumption has been widely accepted in many other researches [19, 20, 26, 29]. As a future work, we will do research on the scenario of variable demands of VMs. The resource demands of the VMs can be achieved by using some prediction techniques, such as [30, 31]. Thus, the server consolidation problem can be depicted as follows: given a current mapping of n VMs to m PMs, how to dynamically remap these VMs to physical servers to optimize some performance criteria, such as the number of active servers and load balance of active servers, under a series of constraints.

4. PROBLEM FORMULATION AND PROPOSED ALGORITHM

4.1 Problem Formulation

In this section, we formulate the server consolidation problem described above. Before giving the optimization objectives, some definitions should be explicitly understood.

Definition 1: *VM Allocation Matrix*: The VM allocation matrix $\mathbf{A}=(a_{ij})_{n\times m}$ is a 0-1 matrix, in which if VM v_i is allocated on physical server p_j, then $a_{ij}=1$; otherwise, $a_{ij}=0$. n and m are the numbers of VMs and physical servers in the system, respectively.

Definition 2: *m-Dimensional State Vector*: $\mathbf{S}=\{s_1, s_2,\ldots, s_m\}$, where $s_j=1$ indicates that physical server p_j is an active server, which means that there is at least one VM running on the sever; otherwise, physical server p_j is in sleep mode or powered off.

Definition 3: *CPU utilization*: the CPU utilization of physical server p_j, denoted by ρ_j, can be defined as the sum of CPU requirement of VMs running on p_j divided by its total CPU capacity [19], *i.e.*,

$$\rho_j = \sum_{i=1}^{n}(a_{ij}\cdot\omega_i)\Big/\Omega_j . \tag{1}$$

Thus, the average CPU utilization of active physical servers, denoted by $\bar{\rho}$, can be given by

$$\bar{\rho} = 1/|\mathbf{S}|^2 \cdot \sum_{j=1}^{m}\rho_j , \tag{2}$$

where $|\mathbf{S}|^2$ is equal to the number of active physical servers, which can be calculated by

$$|\mathbf{S}|^2 = (\sqrt{\sum_{j=1}^{m} s_j^2})^2 = \sum_{j=1}^{m} s_j. \tag{3}$$

Reducing the number of active physical machines in data center to serve the same amount of workloads is of great attraction for cloud operators [29]. Similar to some previous research works [10, 15, 29], in the paper, we also use the number of active physical machines as the main metric to measure the degree of energy consumption in clouds. Then the server consolidation problem becomes how to find a new VM allocation matrix, $\mathbf{A}'=(a'_{ij})_{n\times m}$, which could minimize the number of active servers to reduce energy consumption and balance the load of the active servers to avoid frequent consolidation caused by the unbalance of servers' utilizations. Thus, the multi-objective optimization model can be formulated as follows.

$$\text{Min} \sum_{j=1}^{m} s_j \tag{4}$$

$$\text{Min} \sqrt{1/|\mathbf{S}|^2 \cdot \sum_{j=1}^{m} \left(s_j \cdot (\overline{\rho} - \rho_j)^2 \right)} \tag{5}$$

Subject to

$$\forall i \in \{1, 2, \ldots, n\},\ a_{ij},\ a_{ij}' \in \{0, 1\}. \tag{6}$$

$$\forall i \in \{1, 2, \ldots, n\}, \sum_{j=1}^{m} a'_{ij} = 1. \tag{7}$$

$$\forall j \in \{1, 2, \ldots, m\}, \sum_{i=1}^{n} a'_{ij} \cdot \omega_j \le \Omega_j. \tag{8}$$

$$\forall j \in \{1, 2, \ldots, m\}, \sum_{i=1}^{n} a'_{ij} \cdot \gamma_j \le \Gamma_j. \tag{9}$$

$$\forall j \in \{1, 2, \ldots, m\}, \text{if } \sum_{i=1}^{n} a'_{ij} \ge 1, \text{then } s_j = 1; \text{ otherwise, } s_j = 0. \tag{10}$$

The optimization objectives are minimizing the number of active physical servers, Eq. (4), and minimizing the standard deviation of the CPU utilizations of the active servers, Eq. (5). In Eq. (5), the term s_j indicates whether physical server p_j is an active server or not. If p_j is active, then $s_j = 1$; otherwise $s_j = 0$. The term $|\mathbf{S}|^2$ denotes the number of active physical servers, which is calculated by Eq. (3). Eqs. (6)-(10) are the constraints of the proposed optimization model. Constraints Eqs. (6) and (7) indicate that a VM can only be allocated to one physical server. Constraint Eq. (8) is CPU capacity restriction, which ensures that the total CPU requirements of VMs allocated on a physical server should not exceed its CPU capacity. Constraint Eq. (9) is memory capacity restriction, which is similar to CPU capacity restriction. The last constraint denotes that if there is one or more

VMs allocated on a physical server, the physical server is in active mode, which means that it is an active server; otherwise, the physical server is in sleep mode and it is not an active server.

It can be seen that the formulated server consolidation problem, Eqs. (4)-(10), is actually a variant of the Class Constrained Multiple Knapsack problem [32]. Due to the NP-hardness of this problem, approximation algorithms that suffice to find a near optimal solution are more promising [33]. Therefore, we develop a heuristic algorithm called *greedy-based load balance* (*GBLB*) algorithm.

4.2 The GBLB Algorithm

In this section, we present the details of the developed GBLB algorithm, shown in Algorithm 1, which is used to optimize the current VM allocation. The GBLB algorithm consists of two parts: the first part adjusts the VMs on high utilization servers and the second part consolidates VMs on low utilization servers.

Algorithm 1: Greedy-based load balance (*GBLB*) algorithm.

Input: the current VM allocation matrix **A**, VM set **V** and physical server set **P**.
Output: a new VM allocation matrix **A′**.

```
1  A′=A;
2  while there is a physical server, p_j, whose utilization is higher than U_upper do
3     VMset = VM _ Select(p_j);
4     foreach v_i in VMset do
5        select a destination physical server, p_j′= PhysicalServerSelect (v_i);
6        migrate v_i from p_j to p_j′;
7        change the VM allocation matrix A′;
8     End foreach
9  End while
10 LPS= SelectLowestUtilizationServer(P);
11 while all the VMs on LPS can be migrated to other active servers do
12    foreach v′ running on LPS do
13       select a destination physical server, LPS′= PhysicalServerSelect (v′);
14       migrate v′ from LPS to LPS′;
15       change the VM allocation matrix A′;
16    end foreach
17    switch LPS to sleep mode;
18    LPS = SelectLowestUtilizationServer(P);
19 end while
```

For the first part of Algorithm 1 (lines 2-9, Algorithm 1), we adjust the allocations of the VMs on high utilization servers. The basic idea is to set an upper utilization threshold, U_{upper}, for all physical servers, as it is also used in [5]. If a server's CPU utilization exceeds this threshold, the best VM should be migrated from the server in order to avoid the risk of SLA violation. Algorithm 2 shows the selection of the best VM which must satisfy two conditions [5]: first, the utilization of the VM should be higher than the difference between the upper utilization threshold and the server's utilization; second, if

the VM is migrated from the server, the difference between the upper utilization threshold and the new utilization of the server is the minimum among the values provided by all the VMs on the server (lines 1-9, Algorithm 2). If no such a VM exists, the VM with highest utilization will be migrated to ensure that the number of VM migrations is minimum (lines 10-14, Algorithm 2). This process continues until the new utilization of the server below the upper utilization threshold.

Algorithm 2: VMSelect (p_j)

Input: set of VMs running on physical server p_j.
Output: VMs need to be migrated from p_j.
1 sort all the VMs running on p_j by CPU requirement in ascending order, such as v_{j1}, v_{j2}, …, v_{jl};
2 **while** $p_j.utilization > U_{upper}$ **do**
3 **for** $i = 1$ to l **do**
4 **if** $p_j.utilization - Size(v_{ji})/Size(p_j) < U_{upper}$ **then**
5 add v_{ji} to **VMset**;
6 remove v_{ji} from p_j;
7 **break**;
8 **end if**
9 **end for**
10 **if** ($i > l$)
11 add v_{jl} to **VMset**;
12 remove v_{ji} from p_j;
13 $l=l-1$;
14 **end if**
15 **end while**
16 **return VMset**;

For the second part of Algorithm 1 (lines 10-19, Algorithm 1), we consolidate the VMs running on low utilization servers. If the utilization of an active server is low, all the VMs on this server should be migrated to other active servers and then this server becomes an idle server, which can be switched to sleep mode to eliminate the energy consumption. The consolidation of low utilization servers is an iterative process (lines 10-19, Algorithm 1): for each time, the algorithm selects the active server with the lowest utilization, then judges whether all the VMs on the server can be migrated or not. If all VMs can find a destination active server using Algorithm 3, GBLB algorithm will migrate all the VMs from the server and switch this server to sleep mode to eliminate the power consumption; otherwise, the algorithm stops.

Algorithm 3: PhysicalServerSelect (v_i)

Input: set of physical servers **P** and v_i.
Output: a physical server: *ps*.
1 sort the physical servers in ascending order by residual CPU capacity, such as p_1', p_2', …, p_m';
2 **for** j=1 to m **do**
3 **if** $p'_j.utilization + Size(v_i)/Size(p'_j) <= U_{upper}$ **then**
4 $ps = p'_j$;
5 break;

```
    end if
end for
return ps;
```

In the proposed algorithm, we make sure that the utilization of every active server is below the upper threshold to avoid the violation of application performance and also use greedy strategy to make sure that the utilization of every active server is as close to the upper threshold as possible to reduce the number of active servers and balance workload among the active servers. Although we assume that the resource demands of VMs are constant during their runtimes, the proposed algorithm can also be applied to the scenario where VMs' demands are variable, by regularly re-optimizing the VM-to-PM mapping based on the changes of the VMs' load.

5. PERFORMANCE EVALUATION

To investigate the performance of the proposed algorithm, we compare it with other three algorithms proposed in related researches, *i.e.*, dynamic consolidation with migration control (DCMC) algorithm [10], double-threshold VM selection (DTVS) algorithm [5] and optimal VM placement (OVMP) algorithm [26]. These three algorithms are representative ones which have been proposed to conduct server consolidation to reduce energy consumption in cloud data centers. DCMC algorithm does not migrate the VMs that do not change their resource demands. It remaps the VMs with changing resource demands to the server which is selected using best-fit decreasing heuristic. The basic idea of DTVS algorithm is to set lower and upper CPU utilization thresholds for servers and keep the CPU utilizations of active servers between these thresholds. If a server's CPU utilization is below the lower threshold, all VMs on the server should be migrated to other active servers and the server can be switched to sleep mode. If a server's CPU utilization exceeds the upper threshold, some VMs should be migrated to other servers to reduce its utilization. The idea of OVMP algorithm is to migrate all the VMs on the underutilized physical machines whose remaining available memory resource is less than 50% of total memory capacity, and switch off the idle physical machines. According to [5], the lower and upper utilization thresholds are set to 0.4 and 0.85 respectively in the following experiments.

5.1 Experiment Environment

Our experiments are performed on CloudSim toolkit, which is a modern and extensible simulation framework [34]. All the experiments are performed on a Pentium(R) Dual-Core processor with 2.8GHz and 4GB of memory. The physical infrastructure simulated in the experiments is composed of 1000 heterogeneous physical servers, each of which has one CPU core with CPU capacity uniformly distributed within the range of 100-300 MIPS. The memory capacity of each physical server is uniformly distributed over the set of {2G, 4G, 8G, 16G}. Each VM has a different CPU demand uniformly distributed within the range of 20-80 MIPS and a memory demand which is uniformly distributed over the set of {256M, 512M, 1G, 2G}. As described in section 3, this paper

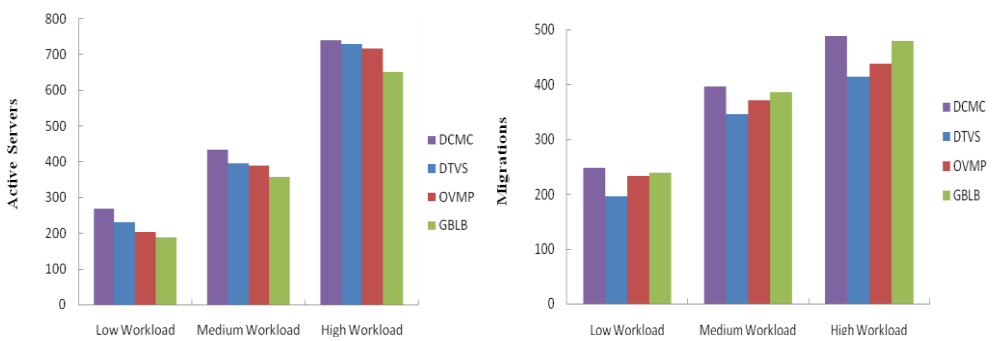
tries to address the problem of server consolidation, which involves the process of dynamically remapping VMs to physical servers. Since runtimes of VMs don't influence this process, we do not consider it in our experiments. To comprehensively evaluate the consolidation performance of our algorithm, we examine three simulated scenarios with different workloads, *i.e.* low workload, medium workload and high workload, and also a real world workload trace [35]. There are 500, 1000, 2000 VMs, which are running in the data center, for low workload, medium workload and high workload respectively. The details of the real world cloud trace are presented in section 5.2.3. Each experiment has been run 2000 times and the results presented in this paper are the mean value of the results obtained by the 2000 experiments.

5.2 Experimental Results

In this section, we present the simulation results of the conducted experiments. In Experiment 1, we compare two performance metrics, *i.e.* the average number of active servers and the average number of migrations, which are also used in [10]. In Experiment 2, we compare two metrics, *i.e.*, the standard deviation of CPU utilizations of all active servers and the average CPU utilization of all active servers. In Experiment 3, we evaluate the proposed algorithm under the real workload trace.

5.2.1 Experiment 1

Fig. 1 shows the results of the two measured metrics obtained by the four algorithms with different workload groups. As can be seen from Fig. 1 (a), the proposed GBLB algorithm needs the fewest active servers to hold all the VMs. Compared with the DTVS algorithm, the proposed GBLB algorithm can reduce active servers by about 22.2%, 10.9% and 12.3% to process low, medium and high workloads respectively. Fig. 1(b) shows that DTVS algorithm needs the fewest migrations. Compared with the DTVS algorithm, the proposed GBLB algorithm needs more VM migrations by about 17%, 10% and 13% in low, medium and high workload situations respectively. Moreover, compared with the OVMP algorithm, the proposed GBLB algorithm can also obtain the less active servers with a little more migrations. It can be seen that the only disadvantage of the proposed algorithm is that GBLB algorithm needs a little more VM migrations than DTVS algorithm. This is because



(a) The number of active serves with different workloads.

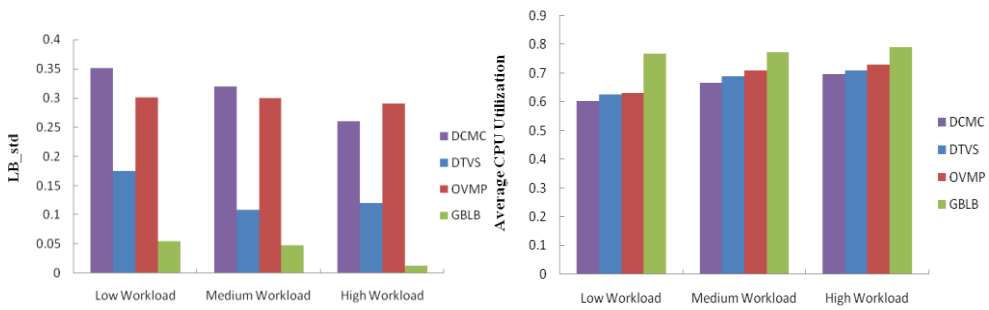
(b) The number of migrations with different workloads.

Fig. 1. Comparisons on the number of active servers and migrations with different workloads.

the proposed algorithm only set an upper utilization threshold for servers and the VMs can be consolidated on as few servers as possible if only the active servers' utilization is less than the upper threshold. Reducing the number of active servers will generally increase the number of virtual machine (VM) migrations. From the comparison, we can see that there has to be a trade-off between the two metrics. In this paper, we treat reducing the number of active servers more importantly than VM migrations because reducing energy consumptions of the active servers is one of the main objectives of this paper. Therefore, the proposed GBLB algorithm can use the fewest active servers to hold all the workloads with only a little more VM migrations.

5.2.2 Experiment 2

In this section, we evaluate the effectiveness of the load balance. The metrics measured are the standard deviation of CPU utilizations of all active servers (*LB_std*) and the average CPU utilization of all active server, $\bar{\rho}$ (Eq. (2)). *LB_std* indicates the discrete degree of the CPU utilizations of active servers and it is calculated by Eq. (4). The smaller the *LB_std*, the better the load balancing level. Fig. 2 shows the results obtained by the four algorithms with different workload groups. It can be easily seen that the proposed GBLB algorithm can achieve the lowest *LB_std*, which indicates that GBLB algorithm can obtain the best load balancing level among the compared algorithms. Also, the average CPU utilization obtained by GBLB algorithm is higher than that obtained by the other three algorithms.



(a) The LB std with different workloads.

(b) The average CPU utilization with different workloads.

Fig. 2. Comparisons on *LB_std* and the average CPU utilization of active servers with different workloads.

To further compare the CPU utilizations of active servers, we randomly choose one result from the 2000 experiments, as it is shown in Fig. 3. For each workload group, a dot represents the CPU utilization of one active server and the straight line represents the average CPU utilization of all active servers. It can be seen that the active servers' utilizations obtained by GBLB algorithm are more even than those obtained by DTVS algorithm for the three different workload groups. This is because GBLB algorithm makes sure that the utilization of every active server is as close to the upper threshold as possible to balance the workload among the active servers.

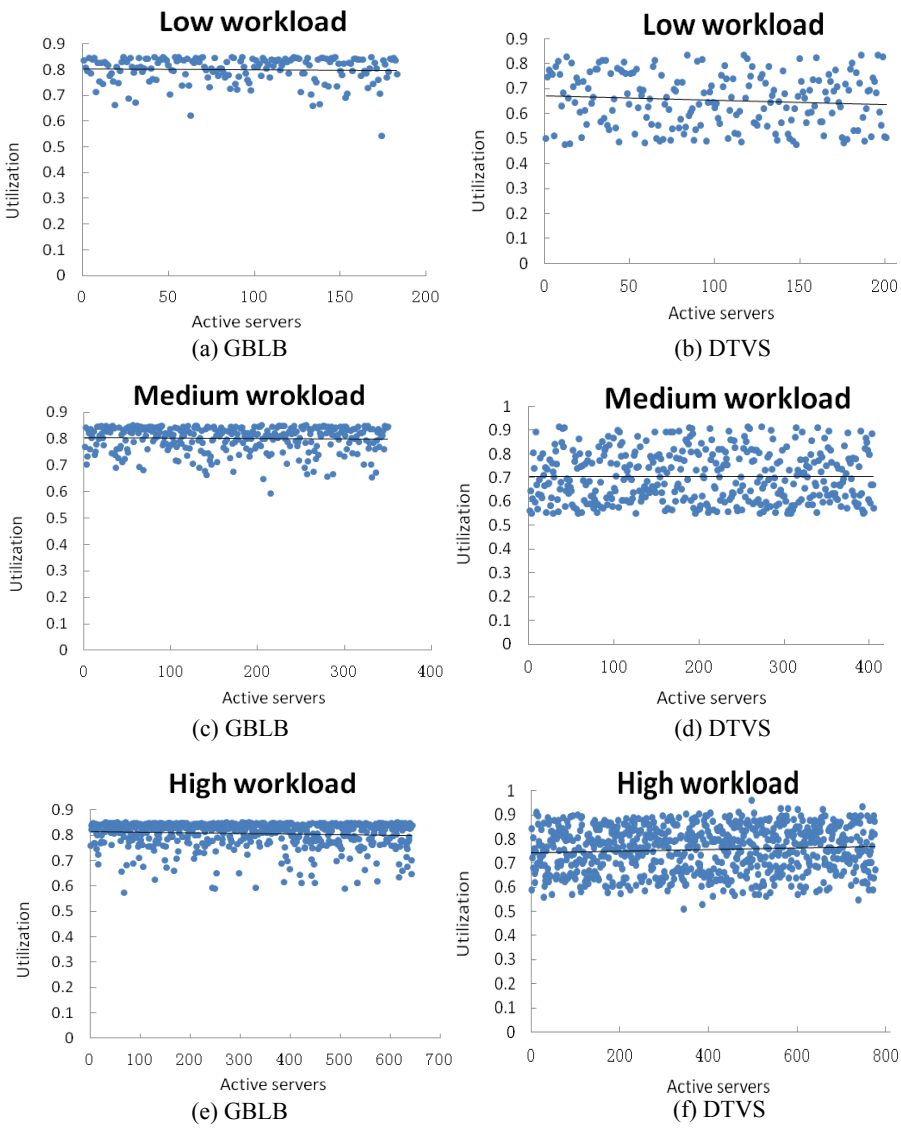


Fig. 3. Active servers' utilization of GBLB algorithm and DTVS algorithm with different workload groups.

5.2.3 Experiment 3

In this section, we compare the performance of the proposed GBLB algorithm with the other two algorithms under Google cluster-usage traces [35], which is generated by the logs from the Google cloud computing cluster. The reported Google cluster-usage traces for version 2.1 contain the records from a cluster of about 12.5k machines over about a month-long period in May 2011. We generally choose 1000 task records from a task event table and 200 machine records from a machine event table. Each task record contains attributes of job ID, CPU request and memory request. The latter two attributes

are normalized values from 0 to 1 (1 represent the maximum value of the CPU/memory request in the task event table). Each machine record contains resource ID, CPU and memory capacity, which are also normalized from 0 to 1 (1 is the maximum CPU/memory capacity in the machine event table). These normalized values are directly used in the experiment because the CPU/memory capacity of machines is about 4-10 times as large as the CPU/memory request. Table 1 shows the results of the four considered metrics obtained by different algorithms under Google cluster-usage traces. It can be seen that the proposed GBLB algorithm can achieve the minimum active servers, best load balance level and highest average CPU utilization. The only disadvantage of GBLB algorithm is that it needs a little more VM migrations than the other two algorithms. Therefore, we could conclude that the proposed GBLB algorithm can also perform well under the real workload trace.

Table 1. Results obtained by different algorithms under real workload trace.

Algorithms	No. of active servers	No. of migrations	LB_std	Average CPU utilization
DTVS	191	**241**	0.085	0.706
OVMP	177	270	0.187	0.735
GBLB	**152**	287	**0.062**	**0.799**

6. CONCLUSIONS

With the increasing demand for computational resources, the number of cloud computing servers has significantly grown and caused enormous amounts of electrical energy consumption, resulting in high operational cost and carbon dioxide emissions. One possible approach to deal with this situation is to use the server consolidation approach, which allows data centers to optimize resource usage and reduce electric power consumption by consolidating multiple VMs on a fewer number of physical servers. However, most existing studies on server consolidation rely on eager migrations without considering the factor of load balance. In this paper, we future take into account this factor and propose a multi-objective optimization model, which minimizes the number of physical servers, as well as balances the load among the physical servers. The proposed algorithm adopts minimum VMs selection policy to avoid the performance degradation caused by VM migrations. It is verified that the proposed approach can reduce the number of active servers and achieve best load balancing level at the cost of a little more migrations.

In the proposed optimization model (Eqs. (4)-(10)), two physical resources, *i.e.*, CPU and memory, are considered. The resource of CPU is formulated as an optimization objective (Eq. (5)) and memory resource is formulated as a constraint (Eq. (9)). However, in modern cloud environments, multi-core physical machines are more promising. As a future work, we will do further research on the scenario of multi-core physical machines.

ACKNOWLEDGEMENTS

This work is supported by Sichuan Provincial Project of International Scientific and Technical Exchange and Research Collaboration Programs (Project No. 2016HH0023).

REFERENCES

1. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, and *et al*., "A view of cloud computing," *Communications of the ACM*, Vol. 53, 2010, pp. 50-58.
2. A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency and Computation: Practice and Experience*, Vol. 24, 2012, pp. 1397-1420.
3. T. Mastelic, A. Oleksiak, H. Claussen, I. Brandic, J. M. Pierson, and A.V. Vasilakos, "Cloud computing: survey on energy efficiency," *ACM Computing Surveys*, Vol. 47, 2015, 1-36.
4. J. Cao, Y. Wu, and M. Li, "Energy efficient allocation of virtual machine in cloud computing environments based on demand forecast," in *Proceedings of the 7th International Conference on Grid and Pervasive Computing*, 2012, pp. 137-151.
5. A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Generation Computer Systems*, Vol. 28, 2012, pp. 755-768.
6. X. Liao, H. Jin, and H. Liu, "Towards a green cluster through dynamic remapping of virtual machines," *Future Generation Computer Systems*, Vol. 28, 2012, pp. 469-477.
7. S. Di, D. Kondo, and C. Wang, "Optimization of composite cloud service processing with virtual machines," *IEEE Transactions on Computers*, Vol. 64, 2015, pp. 1755-1768.
8. W. Dargie, "Estimation of the cost of VM migration," in *Proceedings of the 23rd IEEE International Conference on Computer Communication and Networks*, 2014, pp. 1-8.
9. R. W. Ahmad, A. Gani, S. H. A. Hamid, M. Shiraz, A. Yousafzai, and F. Xia, "A survey on virtual machine migration and server consolidation frameworks for cloud data centers," *Journal of Network and Computer Applications*, Vol. 52, 2015, pp. 11-25.
10. T. Ferreto, M. Netto, R. Calheiros, and C. De Rose, "Server consolidation with migration control for virtualized data centers," *Future Generation Computer Systems*, Vol. 27, 2011, pp. 1027-1034.
11. A. Wolke and C. Pfeiffer, "Improving enterprise VM consolidation with high-dimensional load profiles," in *Proceedings of IEEE International Conference on Cloud Engineering*, 2014, pp. 283-288.
12. E. Carlini, L. Ricci, and M. Coppola, "Flexible load distribution for hybrid distributed virtual environments," *Future Generation Computer Systems*, Vol. 29, 2013, pp. 1561-1572.
13. M. Ajit and G. Vidya, "VM level load balancing in cloud environment," in *Proceedings of the 4th IEEE International Conference on Computing, Communications and Networking Technologies*, 2013, pp. 1-5.

14. W. Voorsluys, J. Broberg, S. Venugopal, and R. Buyya, "Cost of virtual machine live migration in clouds: a performance evaluation," in *Proceedings of the 1st International Conference on Cloud Computing*, 2009, pp. 254-265.
15. Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 24, 2013, pp. 1107-1117.
16. H. Liu, H. Jin, C. Xu, and X. Liao, "Performance and energy modeling for live migration of virtual machines," *Cluster Computing*, Vol. 16, 2013, pp. 249-264.
17. T. Kimbrel, M. Steinder, M. Sviridenko, and A. Tantawi, "Dynamic application placement under service and memory constraints," in *Proceedings of the 4th International Workshop on Experimental and Efficient Algorithms*, 2005, pp. 391-402.
18. A. Karve, T. Kimbrel, G. Pacifici, M. Spreitzer, M. Steinder, M. Sviridenko, and A. Tantawi, "Dynamic placement for clustered web applications," in *Proceedings of the 15th International World Wide Web Conference*, 2006, pp. 595-604.
19. C. Tang, M. Steinder, M. Spreitzer, and G. Pacifici, "A scalable application placement controller for enterprise data center," in *Proceedings of the 16th International World Wide Web Conference*, 2007, pp. 331-340.
20. C. Tian, H. Jiang, A. Iyengar, X. Liu, Z. Wu, J. Chen, W. Liu, and C. Wang, "Improving application placement for cluster-based web applications," *IEEE Transactions on Network and Service Management*, Vol. 8, 2011, pp. 104-115.
21. C. C. Lin, Y. C. Syu, C. J. Chang, J. J. Wu, and P. Liu, "Energy-efficient task scheduling for multi-core platforms with per-core DVFS," *Journal of Parallel and Distributed Computing*, Vol. 86, 2015, pp. 71-81.
22. K. Kim, A. Beloglazov, and R. Buyya, "Power-aware provisioning of cloud resources for real-time services," in *Proceedings of the 7th International Workshop on Middleware for Grids, Clouds and e-Science*, 2009, pp. 1-6.
23. N. Bobroff, A. Kochut, and K. Beaty, "Dynamic placement of virtual machines for managing SLA violations," in *Proceedings of the 10th IFIP/IEEE International Symposium on Integrated Network Management*, 2007, pp. 119-128.
24. B. Speitkamp and M. Bichler, "A mathematical programming approach for server consolidation problems in virtualized data centers," *IEEE Transactions on Services Computing*, Vol. 3, 2010, pp. 266-278.
25. M. Sharifi, H. Salimi, and M. Najafzadeh, "Power-efficient distributed scheduling of virtual machines using workload-aware consolidation techniques," *Journal of Supercomputing*, Vol. 61, 2012, pp. 46-66.
26. V. Perumal and S. Subbiah, "Power-conservative server consolidation based resource management in cloud," *International Journal of Network Management*, Vol. 24, 2014, pp. 415-432.
27. R. Buyya, A. Beloglazov, and J. Abawajy, "Energy-efficient management of data center resources for cloud computing: a vision, architectural elements, and open challenges," in *Proceedings of International Conference on Parallel and Distributed Processing Techniques and Applications*, 2010, pp. 6-17.
28. I. Takouna, W. Dawoud, and C. Meinel, "Energy efficient scheduling of HPC-jobs on virtualized clusters using host and VM dynamic configuration," *ACM SIGOPS Operating Systems Review*, Vol. 46, 2012, pp. 19-27.

29. L. Wei, C. H. Foh, B. He, and J. Cai, "Towards efficient resource allocation for heterogeneous workloads in IaaS clouds," *IEEE Transactions on Cloud Computing*.
30. S. Jang, V. Taylor, X. Wu, M. Prajugo, E. Deelman, G. Mehta, and K. Vahi, "Performance prediction-based versus load-based site selection: Quantifying the difference," in *Proceedings of the 18th International Conference on Parallel and Distributed Computing Systems*, 2005, pp. 148-153.
31. M. D. Felice and X. Yao, "Short-term load forecasting with neural network ensembles: A comparative study," *IEEE Computational Intelligence Magazine*, Vol. 6, 2011, pp. 47-56.
32. H. Shachnai and T. Tamir, "On two class-constrained versions of the multiple knapsack problem," *Algorithmica*, Vol. 29, 2001, pp. 442-467.
33. H. Xu and B. Yang, "An incentive-based heuristic job scheduling algorithm for utility grids," *Future Generation Computer Systems*, Vol. 49, 2015. pp. 1-7,
34. R. Calheiros, R. Ranjan, A. Beloglazov, C. D. Rose, and 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, 2011, pp. 23-50.
35. Google cluster-usage traces (version 2), http://code.google.com/p/googleclusterdata/, Google Inc., 2014.

Heyang Xu is a Ph.D. candidate in School of Computer Science and Engineering (SCSE), University of Electronic Science and Technology of China (UESTC), Chengdu, China. He received the B.E. degree in Computer Science from Yunnan Police Officer Academy, Kunming, China, in 2010. His research interests include grid computing and cloud computing. He has published a regular paper in Future Generation Computer Systems, a regular paper in KSII Transactions on Internet and Information Systems and a paper in the 2nd IEEE International Conference on Parallel, Distributed and Grid Computing.

Bo Yang is a Professor and the Deputy Head of Collaborative Autonomic Computing Laboratory, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China. He received the B.Eng. (1995), and the M.Eng. (1998) degrees from Xi'an Jiaotong University, Xi'an, China; and the Ph.D. (2002) from National University of Singapore, Singapore. His research interests include distributed/Grid/ Cloud computing system, data mining, software and system reliability. He has published over 50 research papers in refereed academic journals and conferences. Prof. Yang is on the Editorial Board of six international academic journals. He served as Program Chair of The 8th IEEE International Conference on Dependable, Autonomic and Secure Computing (IEEE DASC 2009); Program Chair of The 2011 International Conference on Cloud and Service Computing (CSC

2011); Program Vice-Chair of The 12th IEEE International Conference on High Performance and Communications (IEEE HPCC 2010); General Chair of The 2010 International Workshop on Knowledge and Data Engineering in Web-based Learning (IWKDEWL'10); and Program Committee Member of over 30 international conferences or workshops. He is a senior member of IEEE, China Computer Federation (CCF), Chinese Institute of Electronics (CIE), and member of CCF Technical Committee on Collaborative Computing (CCF TCCC).