Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 496843, 7 pages
http://dx.doi.org/10.1155/2014/496843

Research Article

Energy-Efficient Scheduling for Tasks with Deadline in Virtualized Environments

Guangyu Du,[1] Hong He,[1] and Qinggang Meng[2]

[1] *School of Mechanical, Electrical & Information Engineering, Shandong University, Weihai, Shandong 264209, China*
[2] *Department of Computer Science, Loughborough University, Loughborough, Leicestershire LE11 3TU, UK*

Correspondence should be addressed to Hong He; hehong@sdu.edu.cn

Received 22 May 2014; Accepted 10 July 2014; Published 25 September 2014

Academic Editor: Chuandong Li

Data centers, as resource providers, take advantage of virtualization technology to achieve excellent resource utilization, scalability, and high availability. However, large numbers of computing servers containing virtual machines of data centers consume a tremendous amount of energy. Thus, it is necessary to significantly improve resource utilization. Among the many issues associated with energy, scheduling plays a very important role in successful task execution and energy consumption in virtualized environments. This paper seeks to implement an energy-efficient task scheduling algorithm for virtual machines with changeless speed comprised of two main steps: assigning as many tasks as possible to virtual machines with lower energy consumption and keeping the makespan of each virtual machine within a deadline. We propose a novel scheduling algorithm for heterogeneous virtual machines in virtualized environments to effectively reduce energy consumption and finish all tasks before a deadline. The new scheduling strategy is simulated using the CloudSim toolkit package. Experimental results show that our approach outperforms previous scheduling methods by a significant margin in terms of energy consumption.

1. Introduction

Nowadays, large-scale data centers take advantage of virtualization technology [1–3] to achieve excellent resource utilization, scalability, and high availability, such as cloud computing. Cloud computing has achieved tremendous success in offering infrastructure/platform/software as a service based on virtualization technology. Virtualized environments provide computing resource to the clients in the form of a virtual machine (VM) which is a software machine implemented on physical machines. VM behaves like a physical machine, such that it could run different operating systems and applications. Due to poor task assignment optimization, current data centers having huge numbers of heterogeneous servers consume and simultaneously waste massive power to execute numerous assigned tasks. Out of the various energy issues, scheduling plays a very important role in successful execution of tasks in virtualized environments. Scheduling seeks maximum utilization of resources by appropriate assignment of tasks to the resources available like CPU, memory, and storage [4–6]. It is necessary for service providers and requesters to devise efficient scheduling.

In virtualized environments such as cloud computing, end-users simply use the available services and pay for the used services without owning any part of the infrastructure. Several criteria determine the quality of the provided service and the duration of this service (makespan), and the consumed energy is among these criteria. As shown in Section 4, energy consumption and execution time are always two opposite variables. That is to say, in general, there exists no solution which is close to the optimal value on both objectives (makespan and energy consumption) at the same time. For this kind of conflicting problems, these compromise solutions are often used to get trade-offs. Therefore, we will tackle the problem in a compromise way which means that we mainly optimize one objective with the second objective maintained at a reasonable value. For the problem in this paper, it can be represented as minimizing the energy consumption under the premise that the makespan is within a threshold value. However, since

finding the optimal makespan is usually NP-hard, we aim to develop an energy-efficient task scheduling algorithm to complete tasks on heterogeneous virtual machines (VMs) in virtualized environments within a certain deadline, and the total energy consumption is minimized at the same time. Numbers of real applications can be modeled as this kind of problem such as parallel picture transmission and real-time iteration procedure of algorithm in real-time multiprocessing systems and environments.

Most of the current energy-efficient scheduling strategies are based on the dynamic voltage scaling (DVS) [7] or dynamic voltage frequency scaling (DVFS) [8, 9] with the changeable voltage and power supply. However, the current processors with available variable voltage/speed have only several discrete voltage/speed settings [10], which mean that the DVS is still in the development stage. In this paper, we mainly focus on the energy issue in task scheduling, particularly on the condition that the processors or VMs have changeless voltage supply. We prove that assigning as many tasks as possible to the VM with small speed is the key strategy of energy-efficient task scheduling and propose a new task scheduling algorithm that takes into account the makespan and energy consumption at the same time. Our new approach investigates the problem of minimizing energy consumption with schedule length constraint on VMs in virtualized environments.

The rest of the paper is organized as follows. Section 2 discusses the related works. Section 3 defines the task scheduling problem of minimizing energy consumption with schedule length constraint. A new strategy for minimizing energy of task assignment is proposed and the task scheduling algorithm based on the strategy is described mathematically in Section 4. Section 5 presents and discusses the simulations and performance analysis. We conclude our work in Section 6.

2. Related Work

Currently, green computing is applied more and more extensively in data centers. A number of new green scheduling algorithms for saving energy and resource have been proposed, such as [11, 12]. In [13], researchers developed energy-efficient algorithms by incorporating DVS and frequency scaling technology to minimize energy consumption. For the tasks with or without precedence, scheduling algorithms proposed in [14] adopted shared slack reclamation on variable voltage/speed processors to minimize energy consumption. Researchers proved in [7] that only when all tasks were executed with the same power (or at the same speed), the total energy consumption for a computer with multiple identical processors is minimal. Some studies also investigated different ways of minimizing the energy consumption of cloud computing [15].

Numbers of studies focused on parallel applications with precedence constraint and algorithms to minimize the makespan [16–21]. The QoS-based workflow scheduling algorithm proposed in [16] tried to minimize the cost of workflow execution under user-defined deadline constraint based on a novel concept called partial critical paths (PCP). The HEFT (heterogeneous earliest finish time) algorithm [17] is a kind of heuristic method based on list scheduling consisting of two phrases: calculating task prioritization and processor selection. Many previous studies [18–20] proved that HEFT could get competitive result with low complexity.

Genetic algorithm (GA) [22] and the other 10 heuristics are implemented and compared in [23] and the results show that genetic algorithm (GA) behaves best in all the tested algorithms for task scheduling problems. Hybrid particle swarm optimization algorithm (HPSO) [24] belongs to the modified particle swarm optimization algorithms and researchers in [25] showed that the HPSO algorithm for task scheduling problem performs competitively in comparison with the GA based algorithm.

Most previous studies on energy consumption of task scheduling are based on homogeneous computing systems [14, 26–28] or single-processor systems [29]. Researchers in [30] extend the work in [14] with AND/OR model applications which focus on shared-memory multiprocessor systems without consideration of communication. In [28], the researchers adopted DVS (i.e., slack reclamation) to develop a system based on linear programming which exploits slack using. Two scheduling algorithms for bag-of-tasks applications on clusters are proposed in [26]. Researchers of [31] proposed an energy-aware scheduling algorithm with a detailed discussion of slack time computation. The problem of energy-aware task allocation for a computational grid with DVS was studied in [32]. Energy-conscious scheduling heuristic (ECS) that takes into account both makespan and energy consumption is devised in [33].

3. Problem Definition

In the world of virtualized environments (such as cloud computing), successful task scheduling is dependent on the effectiveness of techniques used to execute the task. In our definition, the environment is assumed to be hosted in a data center composed of heterogeneous servers which provide resource by VMs. The servers and VMs may have different memory sizes, processing capacities, and failure rates. Similarly, the communication links may have different bandwidths. It is also assumed that computation can be overlapped with communication. The communications among processors are assumed to perform at the same speed on all links without contentions. Let d be the data center comprised of servers $s_1, s_2, \ldots s_s$. Let $s_j = \{\mathrm{vm}_{j1}, \mathrm{vm}_{j2}, \ldots, \mathrm{vm}_{jm}\}$, where $\mathrm{vm}_{j1}, \mathrm{vm}_{j2}, \ldots, \mathrm{vm}_{jm}$ are VMs in the server s_j. Each VM has its own computing capability or speed represented by the number of instructions per second (MIPS).

A parallel application consisting of tasks can be generally represented by a directed acyclic graph (DAG). The vertices of DAG represent the partitioned tasks of the application and the edges of DAG represent precedence constraints among the tasks (if any), as shown in Figure 1. In this paper, we only focus on independent tasks that all the solutions do not contain idle time. Many real world problems can be modeled as a DAG, such as iterative solution of systems of equations, power system simulations, and VLSI simulation programs. A DAG, $G = (N, E)$, consists of a set N of n nodes and a

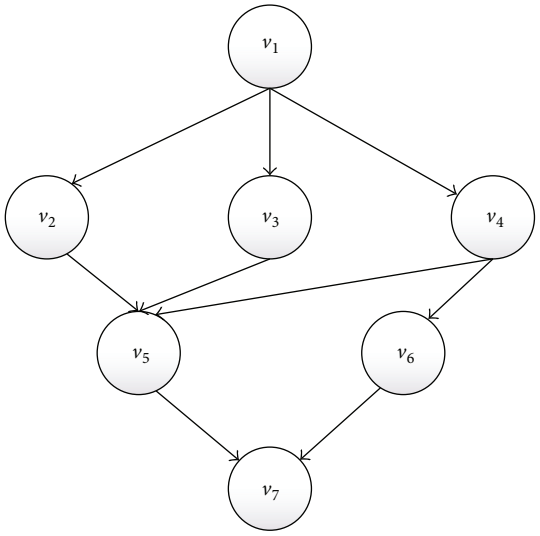

Figure 1: A simple task group.

set E of e edges. Let $V = \{v_1, v_2 \dots v_n\}$, where $v_1, v_2, \dots v_n$ are the sets of tasks to be executed in the data center. Let $r = \{r_1, r_2, \dots r_n\}$, where $r_1, r_2, \dots r_n$ are the sizes of the tasks (execution requirement or the number of instructions).

Because of the dissimilar natures of tasks and VMs, the execution times and energy consumption of a task running on different VMs are different. When a task runs on different VMs, the execution time of it may be different. Similarly, when the communication among two tasks is transmitted through different communication paths, the communication time may be different. Notice that when tasks are assigned to the same VM, the communication cost is zero and thus can be ignored. For ease of discussion, we only consider the independent tasks with no communication.

The scheduling optimization problem of makespan and energy consumption is defined as follows: m VMs in a data center are used to finish n tasks by the deadline time T. Assume that n_i is the number of tasks which are assigned to VM i, for $i = 1, 2, \dots m$; then $n = \sum_{i=1}^{m} n_i$. A changeless speed for each VM is denoted as s_i. The speed is defined as MIPS. The number of instructions of task v_k is denoted as r_k. The execution time for task v_k on VM i is r_k/s_i. The total execution time for n_i tasks on VM i is defined as $T_i = \sum_{k=1}^{n_i} r_k/s_i$. According to [7], the energy consumption for task v_k on VM i is $E_k = r_k s_i^{\alpha-1}$, where $\alpha = 1 + 2/\gamma \geq 3$ for $0 < \gamma \leq 1, i = 1, 2, \dots, m$ and $k = 1, 2, \dots, n_i$. The total energy is $E = \sum_{i=1}^{m} \sum_{k=1}^{n_i} r_k s_i^{\alpha-1}$. The optimization problem is given below.

Minimize $E = \sum_{i=1}^{m} \sum_{k=1}^{n_i} r_k s_i^{\alpha-1}$ with constraints $1 \leq n_i \leq n - m + 1$, $n = \sum_{i=1}^{m} n_i$, $n > m$, $\sum_{k=1}^{n_i} r_k/s_i \leq T$, for $i = 1, 2, \dots, m$, and $k = 1, 2, \dots, n_i$.

Taking into account energy consumption in task scheduling adds another complexity layer to an already complicated problem. Applications in our study are real-time application which means the applications are deadline-constrained. To evaluate the quality of schedules, both makespan and energy consumption should be measured explicitly. Therefore, we consider both makespan and energy consumption as the performance criteria and try to minimize the energy consumption with the makespan constraint. In Section 4, we propose a new energy-efficient task scheduling algorithm that can find an optimal or near optimal schedule to complete all n tasks on m VMs with minimum or near minimum energy E by the deadline T.

4. Energy-Efficient Scheduling with Makespan Constraint

4.1. Energy-Efficient Analysis. As discussed before, the makespan objective is a given hard constraint and we aim at determining the least possible energy consumption. We mainly focus on the biobjective problem to minimize makespan and minimize energy consumption. Unfortunately, these objectives are conflicting. Inspired by [34], we propose Proposition 1 presenting that the minimum energy consumption is obtained only when all the tasks are mapped to the VM which has the minimum speed. However, mapping all the tasks to the VM which has the minimum speed would lead to a schedule that is arbitrarily far from the optimal makespan.

Proposition 1. *Let Schel be a schedule assigning all tasks to VM j_0 (s_{j_0} is minimal) in topological order. Let enc be the energy consumption of the successful execution of schedule Schel. Then, any schedule Schel$'$ $\neq$ Schel, with energy consumption enc$'$, is such that enc $\leq$ enc$'$.*

Proof. Suppose that $j_0 = 0$ (i.e., $\forall j : s_0 \leq s_j$). Let C_0 be the completion time of all the tasks mapped to VM 0; then, $\text{enc} = \sum_{i=0}^{n_0} r_i s_0^{\alpha-1} = s_0^{\alpha} \sum_{i=0}^{n_0} (r_i/s_0) = C_0 s_0^{\alpha}$. Let C_j' be the completion time of the last task on VM j with schedule Schel$'$. Therefore, $\text{enc}' = \sum_{j=0}^{m} C_j' s_j^{\alpha}$. Let N be the task set and L the task sets that are not executed on VM 0 by schedule Schel$'$. Then, $C_0' = C_0 - (\sum_{i \in L} r_i)/s_0$ (there are still some tasks of $N - L$ to be executed on VM 0). Let $L = L_1 \cup L_2 \cup \cdots \cup L_m$, where L_j is task set executed on VM j by schedule Schel$'$ ($\forall j_1 \neq j_2, L_{j1} \cap L_{j2} = \emptyset$). Then, $\forall j, 1 \leq j \leq m, C_j' = (\sum_{i \in L_j} r_i)/s_j$. Let us compute the difference enc$'$ − enc:

$$
\begin{aligned}
&\text{enc}' - \text{enc} \\
&= \sum_{j=0}^{m} C_j' s_j^{\alpha} - C_0 s_0^{\alpha} \\
&= C_0' s_0^{\alpha} + \sum_{j=1}^{m} C_j' s_j^{\alpha} - C_0 s_0^{\alpha} \\
&= s_0^{\alpha} \left(C_0 - s_0^{-1} \sum_{i \in L} r_i \right) + \sum_{j=1}^{m} \left(s_j^{\alpha-1} \sum_{i \in L_j} r_i \right) - C_0 s_0^{\alpha} \\
&= \sum_{j=1}^{m} \left(s_j^{\alpha-1} \sum_{i \in L_j} r_i \right) - s_0^{\alpha-1} \sum_{i \in L} r_i
\end{aligned}
$$

```
{
    Sort all the tasks in the decreasing order of r_i in a waiting list
    Let i = 1
    while the waiting list is not null
        Compute M(i) for task i
        if M(i) is not null then
            Choose the VM j that has the minimum s_j from M(i)
            Assign task i to VM j
            Update the completion date of VM j
            if the completion date of VM j is bigger than the given threshold then
                Mark VM j as non reusable
            end if
        else
            return no solution
    end while
    return the generated schedule
}
```

Algorithm 1: The proposed algorithm for task scheduling problem.

$$= \sum_{j=1}^{m} \left(s_j^{\alpha-1} \sum_{i \in L_j} r_i \right) - s_0^{\alpha-1} \sum_{j=1}^{m} \left(\sum_{i \in L_j} r_i \right)$$

$$= \sum_{j=1}^{m} \left(\left(s_j^{\alpha-1} - s_0^{\alpha-1} \right) \sum_{i \in L_j} r_i \right)$$

$$\geq 0 \quad \left(\text{because } \forall j : \; s_0^{\alpha-1} \leq s_j^{\alpha-1} \right). \tag{1}$$

Proposition 1 presents that assigning all tasks to the VM that has minimal speed could achieve the goal of minimizing energy consumption. On the basis of Proposition 1, we present below an approximation algorithm based on list scheduling which has a lower complexity and is easy to implement.

Let ω be the given makespan (deadline of tasks). Let $M(i) = \{j \mid C_j + r_i/s_j \leq \omega\}$, where C_j is the completion time of the formal last task on VM j. It is obvious that if task i is executed on $j \notin M(i)$, then the makespan will be greater than ω, and $\forall i, i' \in N$ such that $r_i \leq r_{i'}$, $M(i') \subseteq M(i)$. It can be seen from the definition of $M(i)$ that if task i has less operations than task i', then all the machines able to schedule i' with makespan less than ω can also be able to schedule i with makespan less than ω. Notice that if ω is very large, $M(i)$ would contain all VMs and hence all the tasks will be scheduled on the VM with the minimal s_i leading to the most energy-efficient schedule. The proposed approach is illustrated in the proposed algorithm. □

4.2. Algorithm Analysis. The time complexity of the proposed algorithm is in $O(n \log n + m \log m)$. The proposed algorithm could be carried out around a heap. The cost of sorting tasks is in $O(n \log n)$ and sorting VMs costs $O(m \log m)$. The cost of heap operations is in $O(m \log m)$ and scheduling operations costs $O(n)$. The schedule returned by the proposed algorithm could ensure that the makespan is lower than ω or no such schedule exists.

Researchers in [34] proposed a task scheduling algorithm named CMLT which is also a kind of list scheduling. Different from our algorithm, CMLT tackles the reliability of task scheduling problems and it guarantees the makespan is lower than 2ω (not ω) or returns no solution (Algorithm 1).

5. Experiments and Result Analysis

5.1. Experimental Scenarios. In the verification experiments of our algorithm, the comparison experiments were conducted on a PC with a 2.6 GHz Pentium Dual Core Processor, Windows XP platform. Besides, the experiments used CloudSim 3 simulator [35] to simulate virtualized environments. The toolkit of CloudSim 3 simulator supports modeling of virtualized environments like cloud system components such as data centers, host, VMs, and policies of scheduling. Lots of previous studies conducted evaluation experiments based on CloudSim platform.

In our experiments, a set of VMs are created with different speeds (MIPS) using VM components of CloudSim and the RAM size for all the VMs is set to 512 MB. The number of the VM set is fixed at 12. The speed of each VM (MIPS) is chosen uniformly in $[10^2, 10^3]$. The VM set is used for running the task sets to get the makespan and energy consumption results.

To evaluate the performance of the proposed task scheduling algorithm in virtualized environments, randomly generated problem instances are used. We have randomly generated 10 sets of tasks using Cloudlet component where the length of each task (the computation requirement) is chosen uniformly in $[10^5, 10^7]$ and the number of each task set is set from 100 to 1000 with an increment of 100. These numbers may not be very realistic but provide comprehensive results of the tasks that are easy to read. The task sets are scheduled by the proposed algorithm and comparison algorithms when running in the VM set.

The experiment consists of two steps: firstly, the proposed algorithm and comparison algorithms schedule task sets to the VM set to get the makespan and energy consumption; secondly, the makespan and energy consumption of different algorithms are handled and evaluated. The performance of each phase of the proposed algorithm is presented in comparison with the HEFT, GA, and HPSO algorithms, which are three of the best existing scheduling algorithms. As mentioned earlier, the HEFT heuristic algorithm proposed in [17] which is a kind of list scheduling heuristic method could achieve excellent schedule in terms of makespan for independent-constraint tasks. Eleven heuristics are implemented and compared in [23] and the results show that genetic algorithm (GA) behaves best in all the tested algorithms for task scheduling problems. Researchers in [25] showed that the HPSO algorithm for task assignment problem performs competitively in comparison with the GA based algorithm. To evaluate the performance of the proposed algorithm, we implement HEFT, GA, and HPSO as well as the proposed algorithm to compare their performance. Some changes have to be made to the formal HEFT, GA, and HPSO for convenient implementation and comparison. Besides the primary properties of HEFT, GA, and HPSO, we add the constraint of deadlines: given $\forall j \in Q, L_j$ is the task set consisting of tasks assigned to VM j, and then $\sum_{i \in L_j} r_i / s_j \leq \omega$.

The energy consumption of a schedule has been defined earlier: $E = \sum_{i=1}^{n} \sum_{k=1}^{m_i} r_k s_i^{\alpha-1}$, where $\alpha = 1 + 2/\gamma \geq 3$ for $0 < \gamma \leq 1$. The makespan is computed as follows: $M = \max\{\sum_{i \in L_j} r_i / s_j\}$. Moreover, some algorithms such as GA which are stochastic approaches may yield different result with each independent running process; we thus run each algorithm 8 times for every problem instance and report the average results. The average percentage improvement (API) is chosen to conduct the performance analysis of the algorithms. The API in the energy consumption for the proposed algorithm over the HEFT, GA, and HPSO, respectively, is computed as (take HEFT for instance)

$$\left(1 - \frac{\text{Enc}_{\text{the proposed algorithm}}}{\text{Enc}_{\text{HEFT}}}\right) \times 100\%. \quad (2)$$

5.2. Results Analysis. Experiments conducted with different numbers of tasks lead to similar results as shown in Figures 2, 3, and 4. These figures plot the makespan, the energy consumption, and the execution time, respectively, with tasks from 100 to 1000. Figure 2 shows that the proposed algorithm is more efficient than HEFT, GA, and HPSO in terms of energy consumption. In these experiments, we consider schedule length and energy consumption as the main metric. The schedule length of the proposed algorithm for different numbers of tasks is bigger than the other algorithms. This is because the selection of VM order in the proposed algorithm is conducted by giving priority to the VMs with low speed under the threshold makespan constraint, which leads to a greater makespan compared with the results of HEFT, GA, and HPSO algorithms. Notice that the HPSO is slightly better than GA in terms of makespan.

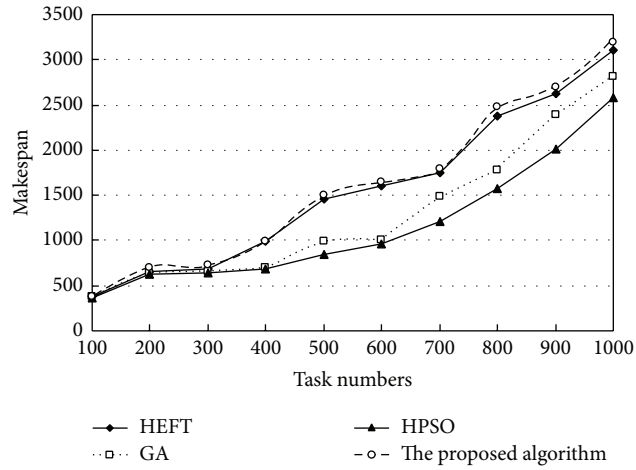

Figure 2: Makespan of HEFT, GA, HPSO, and the proposed algorithm with different numbers of tasks.

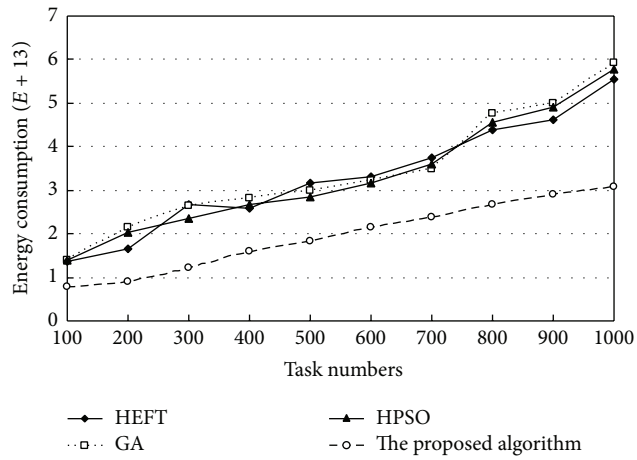

Figure 3: Energy consumption of HEFT, GA, HPSO, and the proposed algorithm with different numbers of tasks.

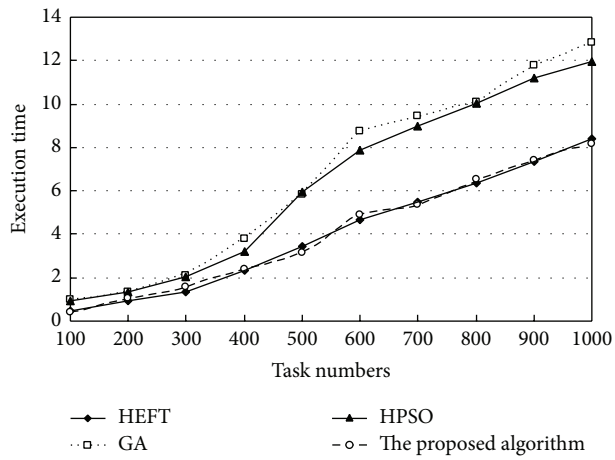

Figure 4: Execution time of HEFT, GA, HPSO, and the proposed algorithm with different numbers of tasks.

Table 1: Computational results for HEFT, GA, HPSO, and the proposed algorithm (denoted as Alg).

N	Enc(E + 13)				API (%)		
	HEFT	GA	HPSO	Alg	/HEFT	/GA	/HPSO
10	1.3778	1.3818	1.3983	0.7786	43.49	43.65	44.32
20	1.6587	2.1356	2.0215	0.9023	45.54	57.75	55.36
30	2.6818	2.6512	2.3426	1.2194	54.53	54.01	47.95
40	2.5874	2.8063	2.6862	1.5984	38.22	43.04	40.51
50	3.1639	2.9899	2.8415	1.8165	42.59	39.25	36.07
60	3.3024	3.2114	3.1632	2.1437	35.09	33.25	33.23
70	3.7576	3.4919	3.6034	2.3796	36.67	31.85	33.96
80	4.3788	4.7745	4.5689	2.6832	38.82	43.80	41.27
90	4.6321	4.9899	4.9128	2.8987	37.42	41.91	40.99
100	5.5504	5.9123	5.7867	3.0896	44.34	47.74	46.61

As Figure 3 shows, the proposed algorithm performs best according to energy consumption indicator. The energy consumption results with different task numbers of the proposed algorithm are much better than those of HEFT, GA, and HPSO. The more detailed results and API are presented in Table 1. As we can see from Table 1, our algorithm is better than metaheuristics (GA and HPSO) and HEFT in terms of energy consumption (e.g., its API with HPSO is from 33.23% to 55.36%). It can be also found from Figure 3 that, for the energy consumption indicator, GA is slightly better than HPSO.

Figure 4 shows the average execution times for HEFT, GA, HPSO, and the proposed algorithm. As the number of tasks increases, the running times of algorithms become longer. As Figure 4 shows, the running times of GA and the HPSO are higher than our algorithm. The running time of HEFT is nearly similar to ours. Moreover, list scheduling heuristics (the proposed algorithm and HEFT) performs better than metaheuristics (GA and HPSO) in terms of running time. The reason for this result is that heuristics based on list scheduling tends to get competitive solutions with low time complexity.

6. Conclusions

This paper has focused on energy-efficient task scheduling on VMs in virtualized environments with changeless variable speed, keeping the makespan of each VM within a deadline. We define the problem of minimizing energy consumption with the constraint of schedule length on VMs in virtualized environments. It has been proved in this paper that the speed of the VMs plays a key role in optimal solutions for energy consumption. Based on the analysis, an energy-efficient task scheduling algorithm has been proposed by combining the list scheduling and the key property of VM speed. Task scheduling on VMs has been implemented in simulation and the results demonstrated the better performance of our algorithm in comparison with three other excellent algorithms (HEFT, GA, and HPSO).

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work is supported by Shandong Province Natural Science Foundation.

References

[1] P. Barham, B. Dragovic, K. Fraser et al., "Xen and the art of virtualization," *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5, pp. 164–177, 2003.

[2] VMware: Virtual Infrastructure Software, http://www.vmware.com/.

[3] A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori, "KVM: the linux virtual machine monitor," in *Proceedings of the Ottawa Linux Symposium (OLS '07)*, pp. 225–230, 2007.

[4] L. M. Zhang, K. Li, and Y. Zhang, "Green task scheduling algorithms with speeds optimization on heterogeneous cloud servers," in *Proceedings of the IEEE/ACM International Conference on Green Computing and Communications (GreenCom '10)*, pp. 76–80, Hangzhou, China, December 2010.

[5] S. Albers, "Energy-efficient algorithms," *Communications of the ACM*, vol. 53, no. 5, pp. 86–96, 2010.

[6] A. Beloglazov and R. Buyya, "Energy efficient allocation of virtual machines in cloud data centers," in *Proceedings of the 10th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGrid '10)*, pp. 577–578, Melbourne, Australia, May 2010.

[7] K. Li, "Performance analysis of power-aware task scheduling algorithms on multiprocessor computers with dynamic voltage and speed," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 11, pp. 1484–1497, 2008.

[8] M. Maurer, V. C. Emeakaroha, I. Brandic, and J. Altmann, "Cost-benefit analysis of an SLA mapping approach for defining standardized Cloud computing goods," *Future Generation Computer Systems*, vol. 28, no. 1, pp. 39–47, 2012.

[9] D. Borgetto, H. Casanova, G. da Costa, and J. M. Pierson, "Energy-aware service allocation," *Future Generation Computer Systems*, vol. 28, no. 5, pp. 769–779, 2012.

[10] G. Qu, "What is the limit of energy saving by dynamic voltage scaling," in *Proceedings of the International Conference on Computer-Aided Design (ICCAD '01)*, pp. 560–563, 2001.

[11] V. T. D. Truong, Y. Sato, and Y. Inoguchi, "Performance evaluation of a green scheduling algorithm for energy savings in cloud computing," in *Proceedings of the IEEE International Symposium on Parallel & Distributed Processing, Workshops and PhD Forum (IPDPSW '10)*, pp. 1–8, 2010.

[12] A. Kamthe and S. Lee, "Stochastic approach to scheduling multiple divisible tasks on a heterogeneous distributed computing system," in *Proceeding of the 21st International Parallel and Distributed Processing Symposium (IPDPS '07)*, pp. 1–11, Long Beach, Calif, USA, March 2007.

[13] Y. C. Lee and A. Y. Zomaya, "On effective slack reclamation in task scheduling for energy reduction," *Journal of Information Processing Systems*, vol. 5, no. 4, pp. 175–186, 2009.

[14] D. Zhu, R. Melhem, and B. R. Childers, "Scheduling with dynamic voltage/speed adjustment using slack reclamation in multiprocessor real-time systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 14, no. 7, pp. 686–700, 2003.

[15] M. Mezmaz, N. Melab, Y. Kessaci et al., "A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems," *Journal of Parallel and Distributed Computing*, vol. 71, no. 11, pp. 1497–1508, 2011.

[16] S. Abrishami and M. Naghibzadeh, "Deadline-constrained workflow scheduling in software as a service Cloud," *Scientia Iranica*, vol. 19, no. 3, pp. 680–689, 2012.

[17] H. Topcuoglu, S. Hariri, and M. Y. Wu, "Performance-effective and low-complexity task scheduling for heterogeneous computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 3, pp. 260–274, 2002.

[18] S. C. Kim, S. Lee, and J. Hahm, "Push-pull: deterministic search-based DAG scheduling for heterogeneous cluster systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 11, pp. 1489–1502, 2007.

[19] Y. C. Lee and A. Y. Zomaya, "A productive duplication-based scheduling algorithm for heterogeneous computing systems," in *Proceedings of International Conference on High Performance Computing and Communications (HPCC '05)*, pp. 203–212, 2005.

[20] D. Bozdag, U. Catalyurek, and F. Ozguner, "A task duplication based bottom-up scheduling algorithm for heterogeneous environments," in *Proceedings of the IEEE International Parallel and Distributed Processing Symposium (IPDPS '06)*, pp. 1–12, 2006.

[21] E. Angel, E. Bampis, and V. Chau, "Low complexity scheduling algorithms minimizing the energy for tasks with agreeable deadlines," *Discrete Applied Mathematics*, vol. 175, pp. 1–10, 2014.

[22] Y. Xu, K. Li, and J. Hu, "A genetic algorithm for task scheduling on heterogeneous computing systems using multiple priority queues," *Information Sciences*, vol. 270, pp. 255–287, 2014.

[23] T. D. Braun, H. J. Siegel, N. Beck et al., "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems," *Journal of Parallel and Distributed Computing*, vol. 61, no. 6, pp. 810–837, 2001.

[24] S. Jiang, Z. Ji, and Y. She, "A novel particle swarm and gravitational search solving economic emission load dispatch problems with various practical constraints," *International Journal of Electrical Power & Energy Systems*, vol. 55, pp. 628–644, 2014.

[25] P. Yin, S. Yu, P. Wang, and Y. Wang, "A hybrid particle swarm optimization algorithm for optimal task assignment in distributed systems," *Computer Standards and Interfaces*, vol. 28, no. 4, pp. 441–450, 2006.

[26] K. H. Kim, R. Buyya, and J. Kim, "Power aware scheduling of bag-of-tasks applications with deadline constraints on DVS-enabled clusters," in *Proceeding of the 7th IEEE International Symposium on Cluster Computing and the Grid (CCGrid '07)*, pp. 541–548, Rio de Janeiro, Brazil, May 2007.

[27] R. Ge, X. Feng, and K. W. Cameron, "Performance-constrained distributed DVS scheduling for scientific applications on power-aware clusters," in *Proceedings of the ACM/IEEE Super computing Conference (SC '05)*, pp. 34–44, Seattle, Wash, USA, November 2005.

[28] B. Rountree, D. K. Lowenthal, S. Funk, V. W. Freeh, B. R. de Supinski, and M. Schulz, "Bounding energy consumption in large-scale MPI programs," in *Proceedings of the ACM/IEEE Conference on Super computing (SC '07)*, pp. 1–9, November 2007.

[29] X. Zhong and C. Xu, "Energy-aware modeling and scheduling for dynamic voltage scaling with statistical real-time guarantee," *IEEE Transactions on Computers*, vol. 56, no. 3, pp. 358–372, 2007.

[30] D. Zhu, D. Mossé, and R. Melhem, "Power-aware scheduling for AND/OR graphs in real-time systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 15, no. 9, pp. 849–864, 2004.

[31] L. Wang, G. von Laszewski, J. Dayal, and F. Wang, "Towards energy aware scheduling for precedence constrained parallel tasks in a cluster with DVFS," in *Proceeding of the 10th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGrid '10)*, pp. 368–377, Melbourne, Australia, May 2010.

[32] S. U. Khan and I. Ahmad, "A cooperative game theoretical technique for joint optimization of energy consumption and response time in computational grids," *IEEE Transactions on Parallel and Distributed Systems*, vol. 20, no. 3, pp. 346–360, 2009.

[33] Y. C. Lee and A. Y. Zomaya, "Minimizing energy consumption for precedence-constrained applications using dynamic voltage scaling," in *Proceedings of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID '09)*, pp. 92–99, May 2009.

[34] E. Jeannot, E. Saule, and D. Trystram, "Optimizing performance and reliability on heterogeneous parallel systems: approximation algorithms and heuristics," *Journal of Parallel and Distributed Computing*, vol. 72, no. 2, pp. 268–280, 2012.

[35] N. Rodrigo, R. Rajiv, B. Anton, A. Csar, 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, no. 1, pp. 23–50, 2011.