IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 2, February 2018

Cloud Scheduling Optimization Based on Greedy Algorithm for Cloud Simulation

Suganya Gladies A.X[1], Ashika Fathima. M[2], Sanofer Parveen. S[3], Sithara. V[4]

Assistant Professor, Computer Science and Engineering, Dhaanish Ahmed Institute of Technology, Coimbatore, India[1]

Students, Computer Science and Engineering, Dhaanish Ahmed Institute of Technology, Coimbatore, India[2,3,4]

Abstract: Cloud computing provides on-demand computing and storage services with high performance and high scalability. However, the rising energy consumption of cloud data centers has become a prominent problem. Scheduling in cloud is responsible for selection of best suitable resources for task execution, by considering some static and dynamic parameters and restrictions of tasks into consideration. The existing deadline constrained application, meeting the application's deadline requirement is critical, but there is no incentive to finish the application earlier. The proposed introduce a model of task scheduling for a cloud-computing data center to energy-efficient dynamic task scheduling. Budget-constrained greedy scheduling algorithm (BCGS). As a heuristic algorithm, BCGS dynamically estimates task energy by considering factors including task resource demands, VM power efficiency, and server workload before scheduling tasks in a greedy manner. Simulated a heterogeneous VM cluster and conducted experiment to evaluate the effectiveness of BCGS. Simulation results show that BCGS effectively reduced total energy consumption by more than 20% without producing large scheduling overheads. Finally, the simulation is carried out and its efficiency is analysed with existing scheduling algorithms.

Keywords: Cloud Simulation, Dynamic VM Allocation, Budget-constrained, Greedy Algorithm.

I. INTRODUCTION

Cloud computing is an approach of using computing as utility. Relatively new term for representing collection of resources which are shared, scaled dynamically. Based on "pay as you use" model, resources can be used or released whenever needed. This refers to both, applications as service to users and servers in data centers which support those services. Cloud computing is a paradigm of distributed computing to provide the customers on-demand, utility-based computing services. Cloud itself consists of physical machines in the data centers of cloud providers. Virtualization technology is used on these physical machines to run multiple operating systems simultaneously. The primary benefit of moving to Clouds is application scalability. Unlike Grids, scalability of Cloud resources allows real-time provisioning of resources to meet application requirements. Cloud services like compute, storage and bandwidth resources are available at substantially lower costs.

Usually tasks are scheduled by user requirements. New scheduling strategies need to be proposed to overcome the problems posed by network properties between user and resources. New scheduling strategies may use some of the conventional scheduling concepts to merge them together with some network aware strategies to provide solutions for better and more efficient job scheduling. Usually tasks are scheduled by user requirements. Initially, scheduling algorithms were being implemented in grids. Due to the reduced performance faced in grids, now there is a need to implement scheduling in cloud. The primary benefit of moving to Clouds is application scalability. Unlike Grids, scalability of Cloud resources allows real-time provisioning of resources to meet application requirements. This enables workflow management systems to readily meet Quality of- Service (QoS) requirements of applications, as opposed to the traditional approach that required advance reservation of resources in global multi-user Grid environments. Cloud services like compute, storage and bandwidth resources are available at substantially lower costs. Cloud applications often require very complex execution environments. These environments are difficult to create on grid resources. In addition, each grid site has a different configuration, which results in extra effort each time an application needs to be ported to a new site. Virtual machines allow the application developer to create a fully customized, portable execution environment configured specifically for their application.

However, many existing cost minimization approaches do not consider that cloud service charges are based on instance hours or minutes. The integral instance hour increases the difficulty for solving the cost minimization problem. The auto scaling scheduling algorithm is one of the algorithms that aims to minimize the cost by considering integral instance hours. In their algorithm, they assign tasks' local deadlines using the same technique as developed. After assigning local deadlines, they decide the number and the types of virtual machines needed to execute the application.

Energy consumption of a data center constitutes a major operation cost. The energy consumed by these largescale data centers. The increasing energy demand could become a hurdle to data center scalability, let alone the carbon footprint they would leave. An Emerson report estimates that the servers of a data center account for 52% of the total consumed

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 2, February 2018

energy, while the cooling systems account for 38%, and other miscellaneous supporting systems, such as power distribution, account for the remaining 10%. These three different sub-systems of a data center may be optimized for energy efficiency.

II. RELATED WORK

Auto-scaling to minimize cost and meet application deadlines in cloud workflows," A goal in cloud computing is to allocate (and thus pay for) only those cloud resources that are truly needed. To date, cloud practitioners have pursued schedule-based (e.g., time-of-day) and rule-based mechanisms to attempt to automate this matching between computing requirements and computing resources. A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems in this paper, investigate the problem of scheduling precedence-constrained parallel applications on heterogeneous computing systems (HCSs) like cloud computing infrastructures. This kind of application was studied and used in many research works. Most of these works existing algorithms to minimize the completion time (make span) without paying much attention to energy consumption. We existing parallel bi-objective hybrid genetic algorithm that takes into account, not only make span, but also energy consumption.
Cost-optimal scheduling in hybrid iaas clouds for deadline constrained workloads examine this optimization problem in a multi-provider hybrid cloud setting with deadline-constrained and preemptible but non-provider-migratable workloads that are characterized by memory, CPU and data transmission requirements. Linear programming is a general technique to tackle such an optimization problem. At present, it is however unclear whether this technique is suitable for the problem at hand and what the performance implications of its use are cost-efficient scheduling heuristics for deadline constrained workloads on hybrid clouds Current approaches for dynamic provisioning of Cloud resources operate at a per-job level, ignoring characteristics of the whole organization workload, which leads to inefficient utilization of Cloud resources. This paper presents an architecture for coordinated dynamic provisioning and scheduling that is able to cost-effectively complete applications within their deadlines by considering the whole organization workload at individual tasks level when making decisions and an accounting mechanism to determine the share of the cost of utilization of public Cloud resources to be assigned to each user.

Online cost-efficient scheduling of deadline-constrained workloads on hybrid clouds use of hybrid clouds introduces the need to determine which workloads are to be outsourced, and to what cloud provider. These decisions should minimize the cost of running a partition of the total workload on one or multiple public cloud providers while taking into account the application requirements such as deadline constraints and data requirements.
A heuristic placement selection of live virtual machine migration for energy-saving in cloud computing environment The field of live VM (virtual machine) migration has been a hotspot problem in green cloud computing. Live VM migration problem is divided into two research aspects: live VM migration mechanism and live VM migration policy. In the meanwhile, with the development of energy-aware computing, we have focused on the VM placement selection of live migration, namely live VM migration policy for energy saving. In this paper, a novel heuristic approach PS-ES is presented. Its main idea includes two parts. One is that it combines the PSO (particle swarm optimization) idea with the SA (simulated annealing) idea to achieve an improved PSO-based approach with the better global search's ability. The other one is that it uses the Probability Theory and Mathematical Statistics and once again utilizes the SA idea to deal with the data obtained from the improved PSO-based process to get the final solution. And thus, the whole approach achieves a long-term optimization for energy saving as it has considered not only the optimization of the current problem scenario but also that of the future problem.

The existing minimizing communication overhead in virtualized computing platforms using decentralized affinity-aware migration decentralized affinity-aware migration technique that incorporates heterogeneity and dynamism in network topology and job communication patterns to allocate virtual machines on the available physical resources. Our technique monitors network affinity between pairs of VMs and uses a distributed bartering algorithm, coupled with migration, to dynamically adjust VM placement such that communication overhead is minimized. A location selection policy of live virtual machine migration for power saving and load balancing novel approach MOGA-LS, which is a heuristic and self-adaptive multi-objective optimization algorithm based on the improved genetic algorithm (GA). This paper has presented the specific design and implementation of MOGA-LS such as the design of the genetic operators, fitness values, and elitism. We have introduced the Pareto dominance theory and the simulated annealing (SA) idea into MOGA-LS and have presented the specific process to get the final solution, and thus, the whole approach achieves a long-term efficient optimization for power saving and load balancing.
Scheduling is that allocating resources to the needed jobs, allocating resources according to the budget constraints, etc., in cloud environment. There are many types of scheduling algorithms available in cloud computing. To achieve high performance, efficient use of resources, best system throughput, budget constraints, Quality of Service (QoS) etc., should be considered. Job scheduling algorithms in cloud computing can be categorized into two main groups; Batch Mode Heuristic Scheduling Algorithms (BMHA) and Online Mode Heuristic Algorithms (OMHA).

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 2, February 2018

III. PROPOSED APPROACH

The greedy approach is very much suitable for those heterogeneous cloud resource environments which are quite dynamic in behaviour and are connected to a process schedule. Greedy algorithm is suitable for dynamic heterogeneous resource environment connected to the scheduler through homogeneous communication environment. Greedy approach is one of the approaches used to solve the job scheduling problem. According to the greedy approach. A greedy algorithm always makes the choice that looks best at that moment. To improve the completion time of tasks greedy algorithm is used with aim of minimizing the turnaround task of individual tasks, resulting in an overall improvement of completion time.

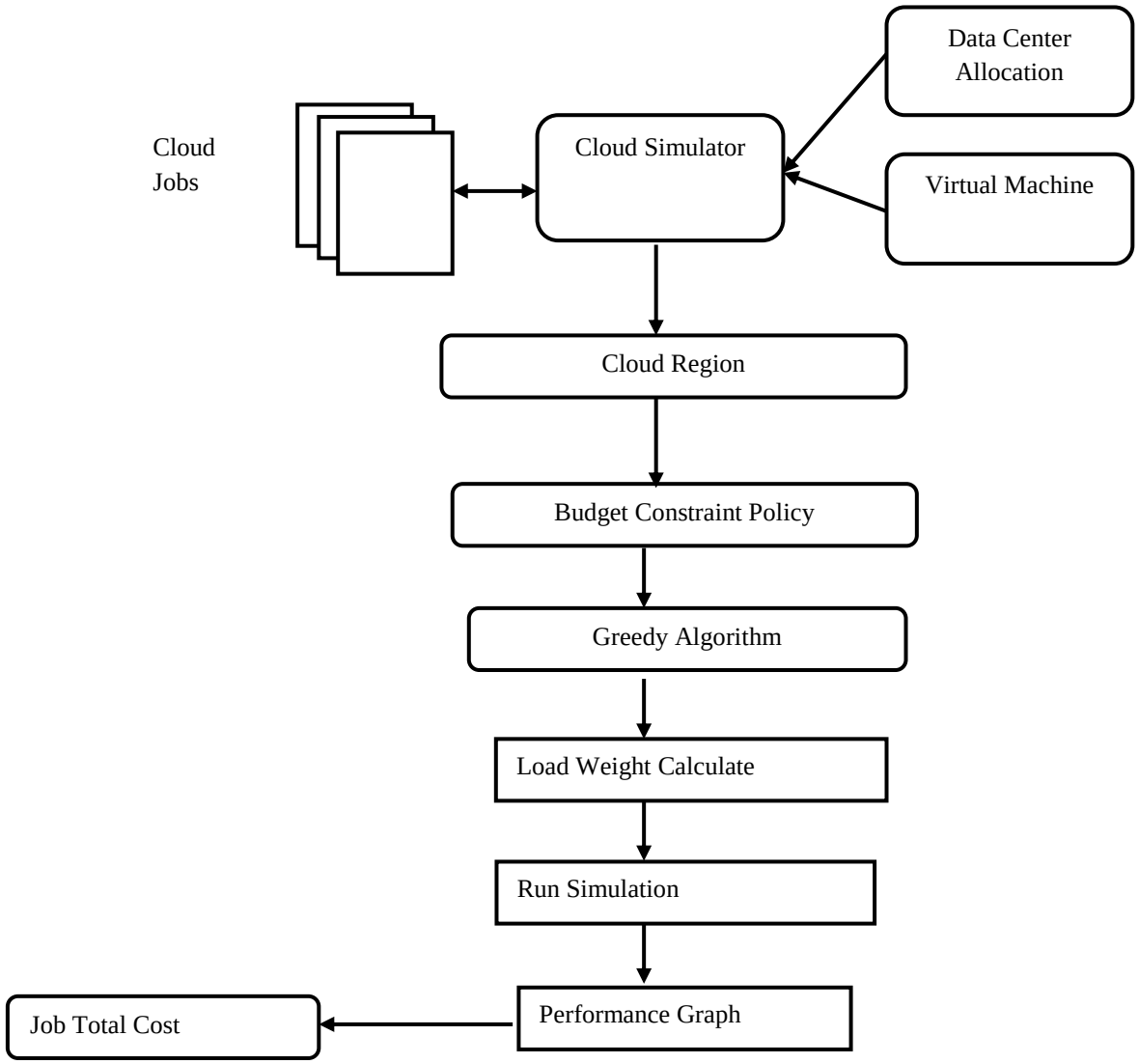

Fig 1. Proposed architectural diagram

A. Cloud Model

A workflow is modelled as a directed acyclic graph (DAG), where each node in the DAG often represents a workflow task, and the edges represent dependencies between the tasks that constrain the order in which tasks are executed. Dependencies typically represent data-flow dependencies in the application, where the output files produced by one task are used as inputs of another task. Each task is a computational program and a set of parameters that need to be executed.

These Virtualization technologies allow the creation of multiple virtual hosts on any of the available servers. There for a task can be flexibly assigned to any server. Servers can be modelled as a system that consumes energy in idle state to perform maintenance functions and to have all the subsystems ready while it waits for task to arrive. Once a task arrives, a server processes the task and it may spend an additional amount of energy, which depends on the number of resources demanded by the task, it is represented as resource utilization in work load model.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 2, February 2018

B. Budget-constrained

In an IaaS cloud, virtualization makes physical resources "transparent" as the applications are run in VMs. To some extent, virtual machine provides independent runtime environment and it is also the basic unit allocated to user applications. In the proposed framework, the energy estimate module predicts the expected task energy consumption on each available VM and sends the data to the scheduler. For energy estimation, the required information includes task resource demands and the power efficiency of each VM.

Job submitted to the cloud will first be decomposed into several tasks. The decomposition principle can be data-based or function-based. Practically, total number of instructions and I/O data size can be estimated by analysing the submitted code or exploiting other existing techniques. Actually, there are many ways to estimate the resource demands of a task. The same job is usually similar. In this paper, we use four "static" attributes to profile a task: number of instructions, the size of data through disk input/output, the size of data through network transmission, and job_id indicating the job it is generated from. The values of these attributes remain unchanged despite the decisions of the scheduler. On the contrary, "dynamic" attributes, including the execution time and energy consumption of a task, are dependent on the features of the VM that executes it.

The proposed the following model for power and energy of CPU in the cloud:

$$P(u) = k \times P_{max} + (1-k) \times P_{max} \times u \quad (1)$$

where isP_{max} the maximum power consumed when the server is fully utilized, k is the fraction of power consumed by the idle server, and u is the CPU utilization. The utilization of CPU may change over time due to workload. Thus, the CPU utilization is a function of time and is represented by u(t). Therefore, the total energy consumption by a physical host can be calculated as an integral of the power consumption function over a period of time:

$$E = \int P\big(u(t)\big) \quad (2)$$

To the best of our knowledge, no research has been carried out on the measurement of the context switch cost, as well as modelling energy consumption of time-shared policy in the CloudSim.

Energy consumption by computing hosts in data centers consists of that of CPU, disk storage, and network interfaces. A strong linear relationship exists between the system CPU utilization and total power consumption of the system. This work has focused on measuring and modelling CPU energy consumption in time-shared policy.

The dynamic power consumption of cloud data centers is mainly produced from the workload on each running server, while the resource demands of tasks are the major sources that drive server workloads. In cloud environment, the demands of tasks can be generally modelled by the task attributes mentioned above. However, it is very difficult to precisely predict the workload as a whole because actually a server has several components (e.g., CPU, memory, disk, and NIC) that keep producing static (idle) and dynamic power. Thus, a possible way is to consider the workload of each component separately. We adopted this ideology and propose to calculate separately the power of computing, storage accessing, and communicating. Particularly in this paper we take the load of the whole server into account and use it to model performance loss.

C. Energy aware greedy scheduling algorithm

The cloud data centers and the increase of computing demands from users, it is of great significance to consider the heterogeneity of both infrastructures and task demands. cloudsim only cast their sight on VM consolidation because it is an effective way to reduce wasted energy by controlling the workload on servers. However, if much load is imposed on servers with low power efficiency, it will cause higher energy cost to warrant the QoS of tasks, which is the situation that service providers are unwilling to face.

A feasible and effective solution is to consider power efficiency in task scheduling. In virtualized environment, collocated VMs can be regarded to have equal power efficiency, which can be calculated by applying (2). Thus, assuming that the infrastructure supports VM procreating and delayed shutdown, propose a Dynamic power aware greedy scheduling algorithm (BCGS). The algorithm takes VM power efficiency and task demands into account and provides a sort of energy-saving task scheduling.

Input: V,M , Q
Output: task-to-VM Mapping

```
(1) Initialize Buffer
(2) Initialize min_energy = MAX_FLOAT
(3) while Q is not empty do
(4)   for i=1 to min{size(Q),buf_size} do
(5)     t= dequeue(Q)
(6)     add t into Buffer
(7)   end
(8)   while Buffer is not empty do
```

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 2, February 2018

(9) for each task t in Buffer do
(10) for each VM k in V do
(11) calculate task_energyt,k
(12) if $task\_energy_{t,k} < \min\_energy$ then
(13) $\min\_energy = task\_energy_{t,k}$
(14) $selected\_task = t$
(15) $selected\_VM = k$
(16) end if
(17) end for
(18) end for
(19) assign selected_task to selected_VM
(20) remove task t from Buffer
(21) update the states V of and M
(22) end while
(23) end while
(24) return Mapping

The heuristic and takes the estimated task execution energy as the evaluation function. We exploit to estimate the execution energy consumption($task\_energy_{t,k}$)of task k on VM k , considering VM efficiency, efficiency loss caused by virtualization, and the performance loss caused by high server workload. Since we adopt task buffer, the process of scheduling is similar to Min-Min and RASA. In other words, the program attempts to search the buffer for a (t^*, k^*)satisfies

$$task\_energy_{t^*,k^*} = \min\{task_{energy\ t^*,k^*}\} \quad (3)$$

where t=0,1…. (buf_size-1) and k=0,1…. n.n is the number of VMs currently available. Then in this round, the scheduler assigns task t to VM k. The pseudocode of VPEGS is shown in Algorithm 1.

IV. EXPERIMENTAL RESULTS

We implemented Dynamic power aware greedy scheduling algorithm (BCGS) and evaluated it in a simulated environment. We also implemented deadline-based scheduling in order to compare their effectiveness. The algorithms and test programs were written in Java (JDK version 1.8.0_65). The simulation was run on a PC equipped with a dual-core Pentium CPU (2.10 GHz) and 4.0 GB memory.
As the targeted system is a cloud computing environment, it is essential to evaluate it on a large-scale infrastructure. Hence, a data center with 100 heterogeneous physical hosts was simulated. Each host was modelled to have a dual core CPU; the performance of each core thereof is equivalent to 1000 million instructions per second (MIPS), 4 GB of RAM, 2 MB of cache memory, and 1 TB of storage. The power consumption by the hosts was defined according to the model described in the previous section. Based on this model, a host consumes power from 210 W with 0% CPU utilization up to 300 W with 100% CPU utilization. Each VM requires one CPU core with 250 MIPS, 128 MB of RAM, and 1 GB of storage. The users submit requests for the provisioning of 10–100 heterogeneous VMs. To model the CPU utilization, each VM runs a web application that uses a uniformly distributed random variable workload and requires 10,000–20,000 MIPS. The results are based on the mean value of running each experiment 5 times.

TABLE I RESOURCE UTILIZATION FOR CLOUD PROVIDER

Algorithm	**Simulation Time**								
	1	5	10	15	20	25	30	35	40
CRED	9.4	10.4	10.8	11.3	11.6	11.6	11.8	11.8	11.8
BCGS	8	8.5	9.2	9.6	9.9	10.1	10.4	10.5	10.5

The evaluation is obtained from the DAG-based applications benchmark provided by Pegasus Workflow Generator. We use four sets of applications from the benchmark, i.e., CyberShake, Laser Interferometer Gravitational Wave Observatory (LIGO), Epigenomics (GENOME), and Montage.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 2, February 2018

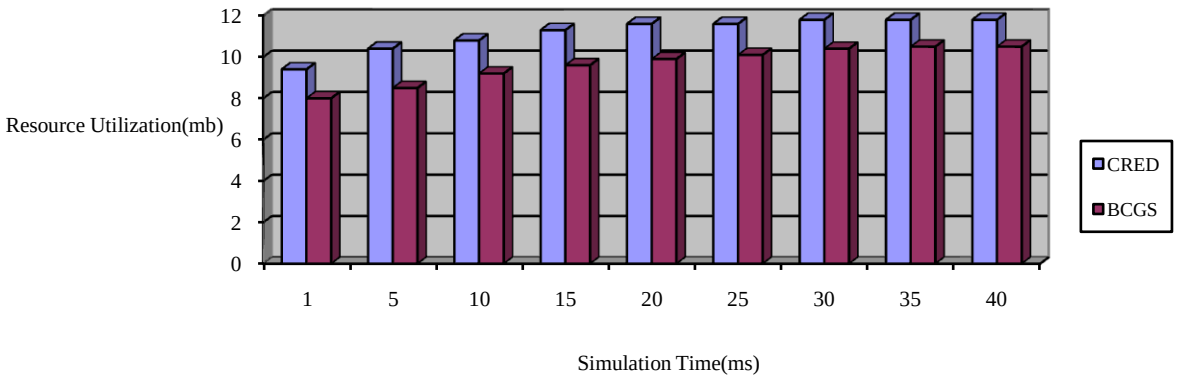

Fig. 2 Compare resource Utilization for cloud provider

The CyberShake applications are highly paralleled applications. The LIGO applications are also highly paralleled, however, they have some critical nodes that have large number of child tasks and parent tasks. Both Epigenomics and Montage applications are combined with parallel execution tasks and sequential tasks. Each set of applications we use for evaluation contains applications with number of tasks ranging from 50 to 1000.

TABLE II POWER CONSUMPTION FOR CLOUD PROVIDER

Algorithm	**Simulation Time**								
	1	5	10	15	20	25	30	35	40
CRED	17	21	27	34	45	51	59	75	89
BCGS	5	8	10	22	33	43	52	68	71

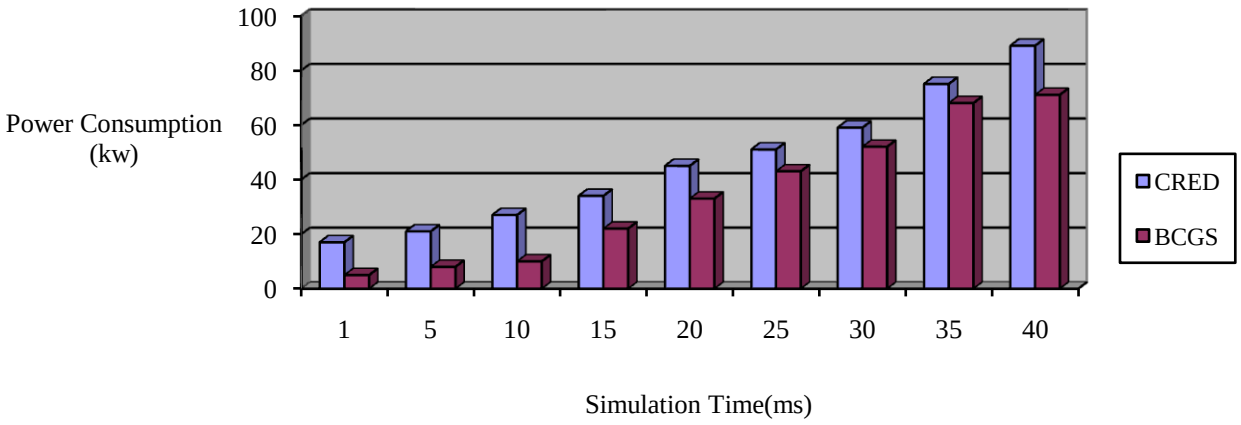

Fig. 3 Comparison of power Consumption for cloud provider

TABLE III VM ENERGY CONSUMPTION RATE FOR CLOUD PROVIDER

Algorithm	**Time**								
	1	5	10	15	20	25	30	35	40
CRED	0.23	0.27	0.31	0.36	0.42	0.53	0.57	0.63	0.67
BCGS	0.14	0.19	0.23	0.27	0.31	0.38	0.43	0.45	0.54

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 2, February 2018

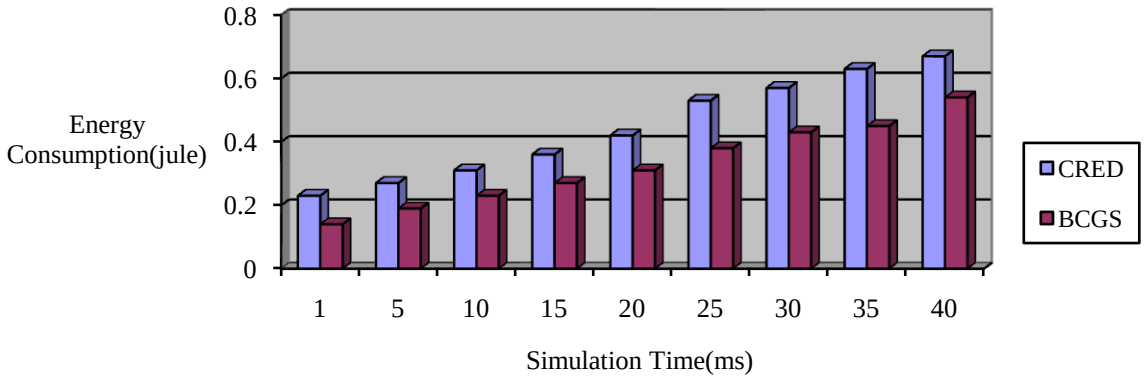

Fig. 4 Comparison of energy Consumption for cloud provider

Algorithm	Cloudlet								
	1	5	10	15	20	25	30	35	40
CRED	253	286	298	318	397	432	497	535	588
BCGS	150	185	210	245	298	345	387	430	550

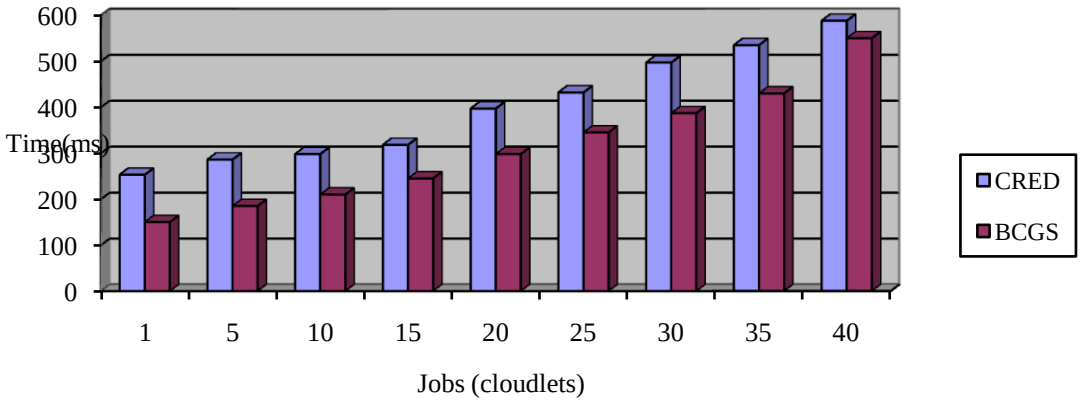

Fig. 4 Comparison of simulation Time vs jobs

V. CONCLUSION

Cloud computing provides on-demand computing and storage services with high performance and high scalability. The existing deadline constrained application, meeting the application's deadline requirement is critical, but there is no incentive to finish the application earlier. The proposed introduce a model of task scheduling for a cloud-computing data center to energy-efficient dynamic task scheduling. Budget-Constrained Greedy Scheduling algorithm (BCGS). As a heuristic algorithm, BCGS dynamically estimates task energy by considering factors including task resource demands, VM power efficiency, and server workload before scheduling tasks in a greedy manner. Simulated a heterogeneous VM cluster and conducted experiment to evaluate the effectiveness of BCGS. Simulation results show that BCGS effectively reduced total energy consumption by more than 20% without producing large scheduling overheads.

REFERENCES

[1] L. F. Bittencourt and E. R. M. Madeira, "Hcoc: A cost optimization algorithm for workflow scheduling in hybrid clouds," J. Internet Serv. Appl., vol. 2, no. 3, pp. 207–227, 2011.
[2] R. N. Calheiros and R. Buyya, "Cost-effective provisioning and scheduling of deadline-constrained applications in hybrid clouds," in Web Information Systems Engineering-WISE 2012, Springer, 2012, pp. 171–184.
[3] F. Ding, R. Zhang, K. Ruan, J. Lin, and Z. Zhao, "A qos-based scheduling approach for complex workflow applications," in Proc. Fifth Annu. ChinaGrid Conf. (ChinaGrid), 2010, pp. 67–73.
[4] J. J. Durillo, R. Prodan, and H. M. Fard, "Moheft: a multi-objective list-based method for workflow scheduling," in Proc. IEEE 4th Int. Conf. Cloud Comput. Technol. Sci., 2012, pp. 185–192.

[5] S. Jayadivya and S. M. S. Bhanu, "Qos based scheduling of workflows in cloud computing," Int. J. Comput. Sci. Electrical Eng., vol. 1, no. 1, pp. 15–21, 2012.
[6] M. Mao and M. Humphrey, "Auto-scaling to minimize cost and meet application deadlines in cloud workflows," in Proc. IEEE Int. Conf. High Perform. Comput., Netw. Storage Anal., 2011, pp. 1–12.
[7] M. Mezmaz, N. Melab, Y. Kessaci, Y. C. Lee, E.-G. Talbi, A. Y. Zomaya, and D. Tuyttens, "A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems," J. Parallel Distrib. Comput., vol. 71, no. 11, pp. 1497–1508,2011.
[8] H. Topcuoglu, S. Hariri, and M.-Y. Wu, "Cost-optimal scheduling in hybrid iaas clouds for deadline constrained workloads," in Proc. IEEE Third Int. Conf. Cloud Comput., 2010, pp. 228–235.
[9] H. Topcuoglu, S. Hariri, and M.-Y. Wu, "Cost-efficient scheduling heuristics for deadline constrained workloads on hybrid clouds," in Proc. IEEE Third Int. Conf. Cloud Comput. Technol. Sci., 2011, pp. 320–327.