**Circuits and Systems, 2016, 7, 1822-1833**
Published Online June 2016 in SciRes. http://www.scirp.org/journal/cs
http://dx.doi.org/10.4236/cs.2016.78157

# Energy Aware Processor Architecture for Effective Scheduling and Power Management in Cloud Using Inclusive Power-Cognizant Processor Controller

**Suma Sira Jacob[1], C. Kezi Selva Vijila[2]**

[1]Department of Computer Science and Engineering, Christian College of Engineering and Technology, Dindigul, India
[2]Department of Electronics and Communication Engineering, Christian College of Engineering and Technology, Dindigul, India
Email: sumasarajacob@gmail.com

Received 26 March 2016; accepted 25 April 2016; published 22 June 2016



## Abstract

**The fast acceptance of cloud technology to industry explains increasing energy conservation needs and adoption of energy aware scheduling methods to cloud. Power consumption is one of the top of mind issues in cloud, because the usage of cloud storage by the individuals or organization grows rapidly. Developing an efficient power management processor architecture has gained considerable attention. However, the conventional power management mechanism fails to consider task scheduling policies. Therefore, this work presents a novel energy aware framework for power management. The proposed system leads to the development of Inclusive Power-Cognizant Processor Controller (IPCPC) for efficient power utilization. To evaluate the performance of the proposed method, simulation experiments inputting random tasks as well as tasks collected from Google Trace Logs were conducted to validate the supremacy of IPCPC. The research based on Real world Google Trace Logs gives results that proposed framework leads to less than 9% of total power consumption per task of server which proves reduction in the overall power needed.**

## Keywords

**Energy Efficiency, Power Management, Task Scheduling, Virtual Machine, Processor Architecture**

**How to cite this paper:** Jacob, S.S. and Vijila, C.K.S. (2016) Energy Aware Processor Architecture for Effective Scheduling and Power Management in Cloud Using Inclusive Power-Cognizant Processor Controller. *Circuits and Systems*, **7**, 1822-1833. http://dx.doi.org/10.4236/cs.2016.78157

## 1. Introduction

Cloud computing and its pay-as-per your use-cost model have enabled the software service providers, application service providers as well as hardware infrastructure service providers and platform service providers to provide computing services on demand and pay per use. This upward drift in cloud computing, combined with the demands for data storage virtualization is driving the rapid evolution of datacenter technologies towards more cost-effective, user driven and energy efficient solutions. Cloud computing is defined as "A large scale distributed computing paradigm that is driven by economies of scale, in which a pool of abstraction, virtualization, dynamically-scalable, managed computing power, storage, platforms, and services are delivered on demand to external customer over the internet" [1]. Power consumption is one of the prominent issues in cloud [2]. In cloud model, data owned by a user is managed in a distributed manner. It will consume more energy for allocating resource to correctly identified user process in a distributed cloud system. Moreover, multiple users access the cloud at same time, and this leads to increase in the energy cost enormously and this high energy consumption produces huge amount of heat, consequently the hardware system fails [3].

In cloud data center due to varying workloads, it is common that most servers run at low utilization. In a cloud datacenter, the energy efficiency can be achieved by making the idle server to sleep thereby by reducing the power consumption. In a low load condition, the processor utilization is 10% and their power consumption is over 50% of the peak power [4]. In the cloud model, multiple data center applications are hosted on a common set of servers. This permits the application workloads to be consolidated in a small number of servers which are always better utilized. Consolidation can be problematic if it loads maximum workload into minimum no of servers and consequently suffers from performance degradation. Thus reducing the energy consumption of cloud data center is a challenging task .The concept of Green computing has gained much attention recently and it was developed for efficient resource utilization as well as for reduction in energy consumption. The proposed work presents a framework for power management in cloud. The proposed idea for the power management is implemented by calculating how much power and configurations are required for the server to process a task such as uploading a file and after that the task will be scheduled to server which requires a minimum power to process the task. The proposed system inaugurates a novel Inclusive Power-cognizant Processor Controller (IPCPC) for minimizing the power utilization and Inclusive Power-cognizant Processor Controller (IPCPC) integrates with collection of configuration management (CCM), Server/Task Mapping (STM), Anticipating power manager (APM). CCM is used for estimating the server configurations in the data center. Server/Task Mapping (STM) is used for scheduling and task mapping. APM can estimate the current power consumption of the server. Inclusive Power-cognizant Processor Controller (IPCPC) enables the CCM (Collection of Configuration Management) to set the configuration of server. APM can estimate the current power consumption of the server by identifying three major portions of the power consumption, such as power consumption of processor execution, power consumption of the server except for processors, and baseline power consumption of the idle processor. The output of APM is given to the Energy aware Earliest Deadline first algorithm. This scheduling algorithm maps the task to the virtual machine of the server. The unused virtual machine of server and their working frequency can be turned off to reduce the power consumption and extend the prolong life time of the multiple servers. The main objective of the proposed work is as follows.

- Enhance the system performance by using a task scheduling algorithm.
- Minimize the power consumption.

The rest of this paper is organized as follows. Section 2 gives the reviews of previous works in power management and scheduling in cloud. Section 3 introduces the detailed architecture about the proposed work, and in Section 4 the experimental results are analyzed. Conclusions are finally drawn in Section 5.

## 2. Related Work

Energy conservation in cloud computing is attracting a wide range of attention in research area, and is leading to a new computing era known as green computing. Efficient scheduling techniques are there to reduce the energy conservation in data centers which have been thoroughly examined in [5]-[7]. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat, and R. P. Doyle propose the energy-efficient management issue of homogeneous resources in Internet hosting centers. The proposed method is ideal for power efficient resource allocation at data center level and energy consumption is reduced by switching idle servers to power saving modes [8]. Arindam Banerjee, Prateek Agrawal, N.Ch.S.N. Iyengar [2] investigate all possible areas in a typical cloud infrastructure that are responsible for significant amount of energy consumption and proposes methodologies for decreasing power utilization.

Shin-ichi Kuribayashi [3] identifies the need of collaboration among the entire servers, the communication network, and the power network for reducing power consumption in cloud environment. This paper proposes to use signaling sequences to exchange the information on power consumption between network and servers. In order to realize the proposed policy the volume of power consumption method by all network devices has been estimated and assigns it to an individual user. Luna Mingyi Zhang, Keqin Li, Dan Chia-Tien Lo and Yanqing Zhang [4], considers several green task scheduling algorithms for heterogeneous computers which will have continuous speeds and discrete speeds. All these algorithms focus on minimizing the consumption of energy as well as determining an optimal speed for the tasks assigned to the computer. Awada Uchechukwu, keqiu Li, and Yanming Shen [9], characterizes energy consumption and performance in cloud environments by analyzing and measuring the impact of various task and system configuration. This paper presents energy consumption formulas for calculating the total energy consumption in cloud environments. Andrew J. Younge, Gregor von Laszewski, Lizhe Wang, Sonia Lopez-Alarcon and Warren Carithers [10], presents a framework for providing efficient green enhancements within the scalable cloud computing architecture. The frame work derives efficient methods for VM scheduling, VM image management, and advanced data center design. The Scheduling technique addressed here contains the placement of VMs within the Cloud infrastructure while minimizing the operating costs of the Cloud itself. This is typically achieved by optimizing either power of the server equipment itself or the overall temperature within the data center. The image management attempts to control and manipulate the size and placement of VM images in various ways to conserve power. Yan Ma, Gong B, Sugihara R, and Gupta R. [11], investigates the power-aware scheduling algorithms for heterogeneous systems to meet the deadline constraints in high performance computing applications. A pricing scheme for tasks is also presented in the way that the price of a task differs as its energy usage and the price of a task will depend on the rigidity of its deadline.

Lizhe Wanga *et al*. [12] studies the case of reducing power consumption of parallel tasks in a cluster with the Dynamic Voltage Frequency Scaling (DVFS) technique. This paper also discusses the relationship between energy consumption and task execution time.

Robert Basmadjian, Hermann De Meer, Ricardo Lent and Giovanni Giuliani [13] studies the case of private cloud computing environments from the perspective of energy saving concerns. This paper presents a generic conceptual description for ICT resources of a data center and identifies their corresponding energy-related attributes. Power consumption prediction models for servers, storage devices and network equipment are presented in this paper and shows that by applying appropriate energy optimization policies guided through accurate power consumption prediction models, it is possible to save about 20% of energy consumption when typical single-site private cloud data centers are considered.

Recently, a number of research works have been conducted in energy efficient scheduling data centers [14]. The orthodox power reduction system in a cloud system agrees on an automatic scheme to control the usage of peripheral operations and processor frequency. These mechanisms fail to meet user requirements, consider workloads and operational status of processors in the multiple cloud servers in a data ware house. Also, the multiple Processors are not required since most of the idle time of cloud devices is not heavy loading. The unused idle processors can be shut down to save more power. In this paper a novel framework is established with the consideration of reduction in total energy consumption in datacenters. The proposed method shows that by applying energy consumption reduction technique and suitable scheduling technique, it is possible to save large amount of power in cloud data centers. Our main contributions on cloud storage by the proposed Inclusive Power-cognizant Processor Controller are as follows.

- Innovative concept to reduce power consumption of server by Server/Task Mapping.
- Power management for the entire cloud storage system.

## 3. Power Aware Processor Using Inclusive Power-Cognizant Processor Controller

This section gives the detailed explanation of the energy aware scheduler IPCPC which is proposed to minimize the power consumption of the server and thus enhances the system performance. IPCPC will collect the configuration details of the server when issuing or completing the task based on the current status of the server and server workload configuration. It can manage host off/on states, adjust the working frequency, and schedules the task queues of each server to achieve best system performance and to reduce the power consumption of the server system. To achieve the above objective, this mechanism schedules tasks of the task set under some con-

straints. First, the tasks entered in to the system will be sorted based on their deadline. Second, all possible system configurations are determined by IPCPC. Then, the tasks are scheduled to most feasible configuration to achieve an improved load balance as well as reduced power consumption. To achieve this IPCPC processor manager uses three techniques CCM, STM, APM. **Figure 1** illustrates the conceptual organization of IPCPC. The following subsections introduce the details of these three mechanisms.

### 3.1. Collection of Configuration Management Technique

In cloud system with IPCPC, the huge number of tasks are submitted into the cloud and these tasks are maintained in task-set which is denoted as Ti = {T1, T2 …, Tn}. Assume the available number of servers of the cloud is denoted as K. The Datacenter has much number of servers and is denoted as Si= {S1, S2, …, Sk}; and each server has number of virtual machines based upon their capacity. The enabling status of server is denoted as Si = 1, when the corresponding server is in power on stage, and S1 = 0, when the server is in shutdown/sleep stage. The set of all possible combinations of the data center enabling status is denoted as DCi = {S1, S2, ...., S(k-1)}, where DC1 = (1, 0, …, 0) and DC(k-1)= (1, 1, …, 1), the number of combination of the server enabling status is 2(k-1). The set of possible working frequencies of server Si is denoted as Fi= {fi j | 1≤ j ≤ m, fi1 < fi2 <… fim}, where f1 is the lowest frequency and fm is a highest frequency. Therefore all working frequencies of the server systems are denoted as Freq = {F1… Fg… Fk}, Fg ∈ {f1…fm | f1 < … < fm}. The workload and executed server number of task i is denoted as Ti.L and Ti.S. The set of all tasks is denoted as TSi = {T1.S, T2.S… Tn.S}, also TS_cur and TS_temp represent the current task set and temporary task set. The proposed CCM technique must be executed to evaluate a feasible server configuration. The CCM is executed when a task is issued or when a task is initiated by Tissue = Begin or when task issue is completed by Tissue = Completed. CCM can determine the possible system configurations, which can achieve the lowest virtual machine migration; excellent load balance and the highest working frequency. From the collected configuration details, a suitable one for allocation is selected. The server system configuration is denoted as,

$$\text{Config} = \{\text{Power}, \text{S}, \text{freq}, \text{L}, \text{TM}\} \tag{1}$$

This can be generated by CCM. Equation (1) consists of five components, where S denotes reasonable server system, freq refers to working frequency of server and the Power denotes expecting power consumption of the

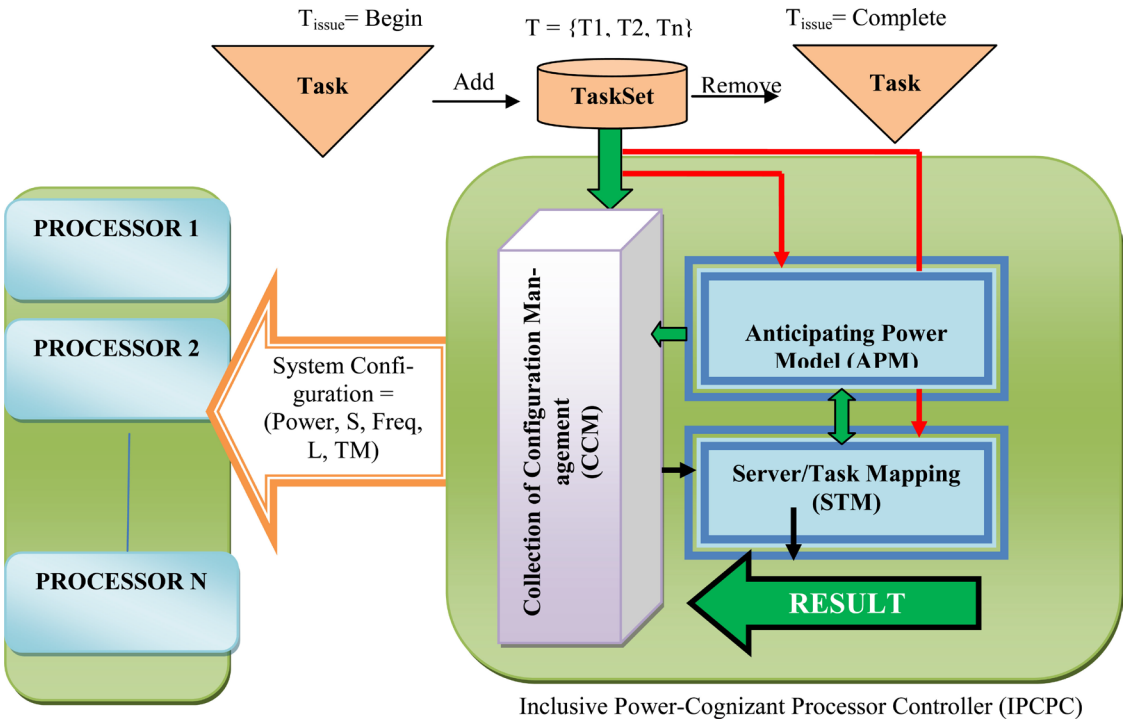


**Figure 1.** The organization of proposed IPCPC system.

server and is calculated using Anticipating power model. L is denoted as highest working load of server and TM is the maximum of task migration number achieved by STM. The additional functions of the IPCPC are listed as follows,

Offline computing evaluates the relevant parameters, $\lambda$, $\omega$, $\beta$ and TS_cur, which are used by CCM and APM.

Server/Task Mapping (STM) (Ti.L, Di) schedules and assigns the tasks based on their load and deadline. The following subsection discusses the technique in detail.

Load (i, TS) estimates and returns the work load value of the task set on the server i, where task set is scheduled and reassigned in order to improve the load balance. This value is also used by the Anticipating power model to predict the power.

## 3.2. Server Task Mapping

The power consumption of a server is notably affected by the workload of server in cloud. As a matter of fact good workload balance among the servers will improve the overall performance of the datacenter. To achieve load balancing the proposed concept uses an effective scheduling algorithm called Earliest Deadline first. The scheduling algorithm considers the following factors such as deadline, cost, reliability and availability of workflow. The performance of job depends on the execution time (ei (Ti)) of task (Ti) which has to be executed on server machine. For this reason execution time of task should be calculated before assigning to server based upon MIPS rate. Deadline of task is represented as di.

### 3.2.1. Task Arrangement

In cloud large number of tasks Ti = {T1, T2, …, Tn} and servers Si = {S1, S2, Sk} are available. The algorithm 1 has detail description of task arrangement in queue. Initially assume the queue Q, current task set (TS_cur); temporary task set (TS_temp) are empty sets. The current task set contains the currently available number of task for scheduling and temporary task set maintains the currently executed tasks. If Task Ti enters into the cloud, here it is mentioned as Tissue = Begin.

Algorithm 1: Task arrangement

1) Q ← Ø; and TS_Cur ← Ø: TS_temp ← Ø
2) Tissue = Begin
3) For each task Ti enter into Q
4) Q ⊂ Ti= {T1, T2, …, Tn}
5) Calculate Total load of the entire tasks
6) di = Deadline of Ti
7) Sort all task Ti based on their load and deadline in ascending order of their length
8) If Tn.L < Tn+1.L
9) Assign Tn.L to Si
10) EDF scheduling ()
11) Tn.L → TS_temp
12) TS_Temp+1

The basic idea of the proposed algorithm for task arrangement is to arrange the arriving task set based on the dead line. The load of each task is calculated as Ti.L using the auxiliary function Load (i, TS) and deadline as di based on the task length, where i, varies from 0 to n. The task set is maintained in queue and the workload of task set is the summation of the individual task load. Each task is sorted in ascending order of their load and their deadline. Flow chart is explained in **Figure 2**.

### 3.2.2. EDF Scheduling Algorithm

1) Task Q ≠ {Ø}.
2) Use CCM to gather configuration of server.
3) Sort all the Si in descending order.
4) If Si has feasible configuration then
5) Choose Si with config = {Power, s, Freq, L, TM}
6) Assign task Ti to Si
7) Update Si.config = {Power, s, Freq, L, TM} after completing allocation of task.

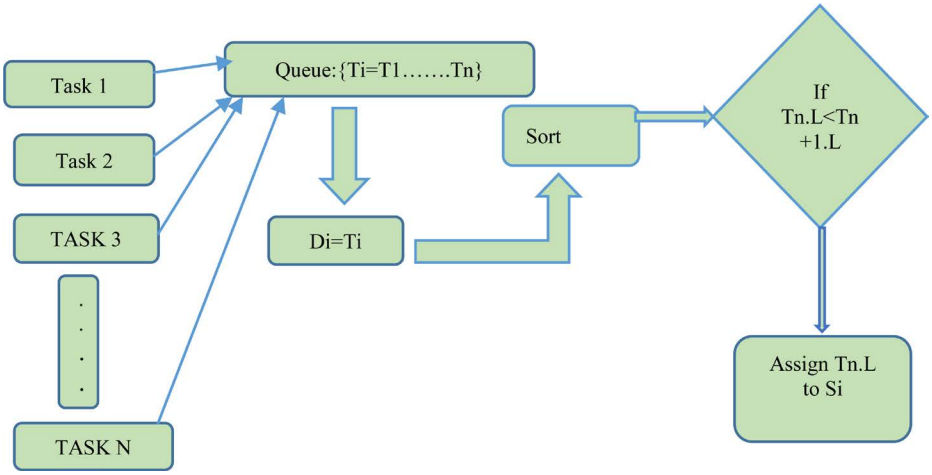


**Figure 2.** The data flow diagram for Task arrangement algorithm.

Else

8) If Si.Config < Si + 1.Config

9) Choose Si +1.Config

The basic idea of the EDF scheduling algorithm is to use the APM and CCM to balance the work load and to reduce the power consumption of severs. Before the task is scheduled calculate the power of current server system by using APM and by using CCM measure the system configuration. Based on this information the task is scheduled to the feasible server system. Then TS_temp will be increased by one. When the task is fully completed then Tissue = Completed is initiated. The algorithm 2 has a detailed description of the scheduling process. EDF scheduling algorithm takes the parameter as the load of individual task, and its deadline. The proposed work calculates the deadline by considering the task length. The process which has minimum load and earliest deadline is sent to the head of the queue. This process is assigned to the enabling server i. The current task in queue is submitted for scheduling after arranging the tasks in ascending order of their workload and deadline. The server system configuration is identified and status of the server is evaluated. Based on this information it is found out whether the server has the capability to accommodate the task, if so, task will be allocated to the server. **Figure 3** shows the data flow diagram for the EDF scheduling algorithm.

### 3.3. Anticipating Power Model

As mentioned earlier, IPCPC has three major techniques to define the power aware model of cloud. The power level of each server can't be calculated exactly and promptly by using power meter. So this section explains how to predict the power of server by using APM. It estimates the current power consumption of server by identifying the three major portions of the power consumption. They are power consumption of server execution, power consumption of the other components except for server and base line power consumption of the idle server.

$$APM = \varepsilon + \beta \tag{2}$$

In Equation (2), APM denotes the predicted power, ε denotes the power consumptions of server's core processor and $\beta$ represents the power consumptions of other components except for server processor in the cloud system. $\beta$ can be treated as constant when the configurations of the components in the cloud server are same. When the data center consist K servers, Power consumption of server is denoted as $\mu$. Equation (3) shows Total power consumption of the Servers is

$$\varepsilon = \sum_{h=1}^{k} \mu h \tag{3}$$

According to the results of [15] [16], power consumption of the server core is formulated as P = KCV2f, K denotes the constant; C represent the capacitance of the server; V refer to the working voltage of the server and f is the working frequency of the server processor. While the system work load is increased, the power consumption of the server processor is also increased. The enabling status of the server ON/OFF state also affects the

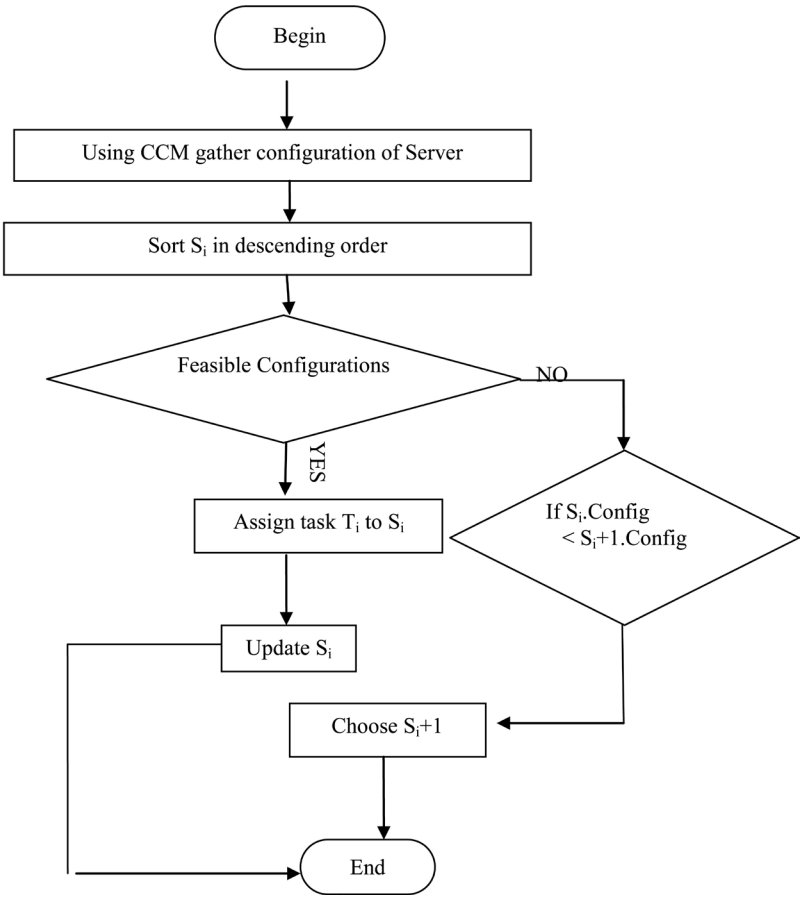


**Figure 3.** The data flow diagram for EDF scheduling algorithm.

power consumption of server. So Equation (3) can be extended as in Equation (4).

$$\mu h = Ph \times C \times Vh2 \times Fh \times \omega \times loadh \tag{4}$$

where *Ph* denotes the enabling status of the server *h*; *Ph* = 1 refers a situation in which the power of the server is turned ON; *Ph* = 0 refers the situation in which the server *h* is OFF/sleep. *Fh* represents the working frequency of the server. *Vh* denotes the working voltage of the server; *loadh* refers to the work load of the server, which can estimate from the additional functions of IPCPC load (i, Ti.L) and TS_cur. Moreover, ω is a constant factor of workload and the power consumption of the server. Finally, the overall power consumption of the system can be represented as Equation (5).

$$APM = \left(\sum_{h=1}^{k} Ph \times C \times Vh2 \times Fh \times \omega \times loadh\right) + \beta \tag{5}$$

where $\varepsilon$, $\beta$, and $\omega$ can be obtained from the offline-computing (), it can be varied based on the various cloud system.

## 4. Simulation and Experimental Results

This section explains the experimental analysis of IPCPC that is defined in section 3. Experiments are conducted to analyze power consumption of each server. So, here to demonstrate the performance improvements of the IPCPC, the proposed algorithm is compared with EARH [17], and also compare with some existing scheduling algorithm like Greedy-R [18], Greedy-P [18], and FCFS [18].

The performance metric, by which the proposed system assesses the system performance, includes following power consumption parameters. The parameter Resource utilization by task (RU) is the number of resources used by a task. Effective utilization (EU) defines whether resources are effectively utilized by varying number of

tasks. Guarantee ratio (GR) gives total number of tasks guaranteed to meet their deadlines from the entire task set. The Total energy consumption (ΔEC total) parameter gives total energy consumed by server and Power consumption per task (PCT) gives total power consumption per accepted task count.

### 4.1. Experimental Setup

Cloudsim tool kit is used as simulation platform in this application. A data center has been simulated comprising multiple hosts with the CPU performance equivalent to 9600 MIPS, 40 GB RAM and 11 Tb of storage. Each Virtual machine requires up to 2400 MIPS. These VMs are needed in order to support a wide variety of hardware, software and varying user tasks. A hypervisor Xen provides the virtualized hardware to each VM. Next there is a need for an operating system within the VM to accomplish the task. X86 hardware is suggested for this application with includes operating system Linux. This configuration is able to detect various load of the task. This is takes only 15 seconds for running mod-probe to load single module.

The aim of this set of experiments is to validate the performance effect of EDF scheduling algorithm. **Figure 4** shows the performance of the EDF scheduling algorithm which is compared with Cura [19] and the other three existing algorithms. The parameter used here to compare is resource utilization with varying deadlines.

To demonstrate the performance improvements of the IPCPC, the proposed algorithm is compared with EARH, and it is also compared with some existing scheduling algorithm like Greedy-R, Greedy-P, and FCFS. **Figure 5** shows the comparison graph between proposed IPCPC with Cura. The parameter considered for

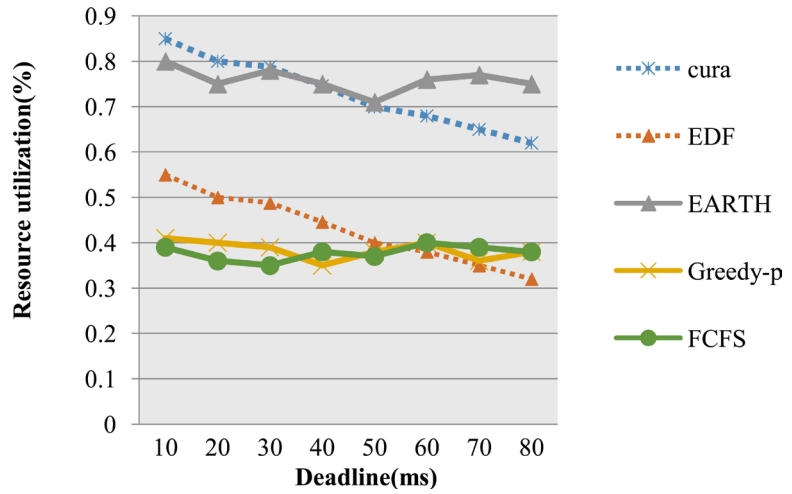


**Figure 4.** Dead line based resource utilization of IPCPC.

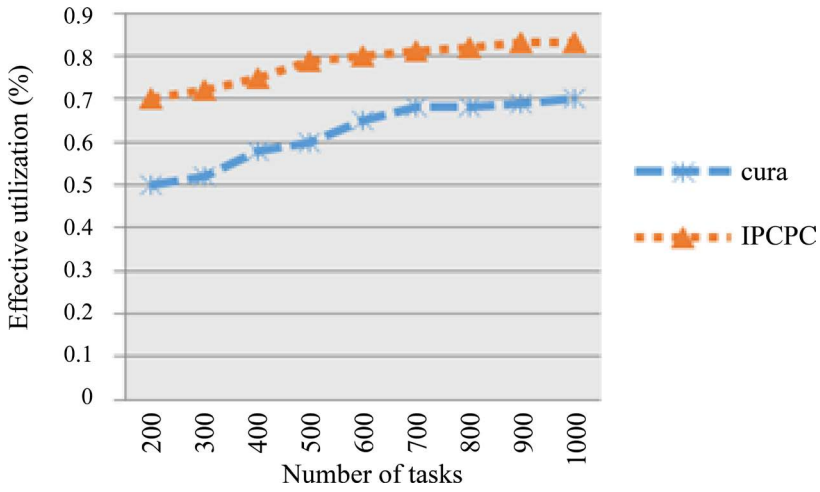


**Figure 5.** Resource utilization based on number of task.

the comparison is resource utilization with varying task count. The resource utilization parameter is considered for comparisons because ineffective utilization of the resources of cloud can definitely leads to diminishing power consumption. The aim of this set of experiments is to validate the performance effect of EDF scheduling algorithm.

**Figure 6(a)** shows the algorithm basically keeps the guarantee ratio even if the value of task count is varied. IPCPC with EDF can have a higher guarantee ratio than other algorithm.

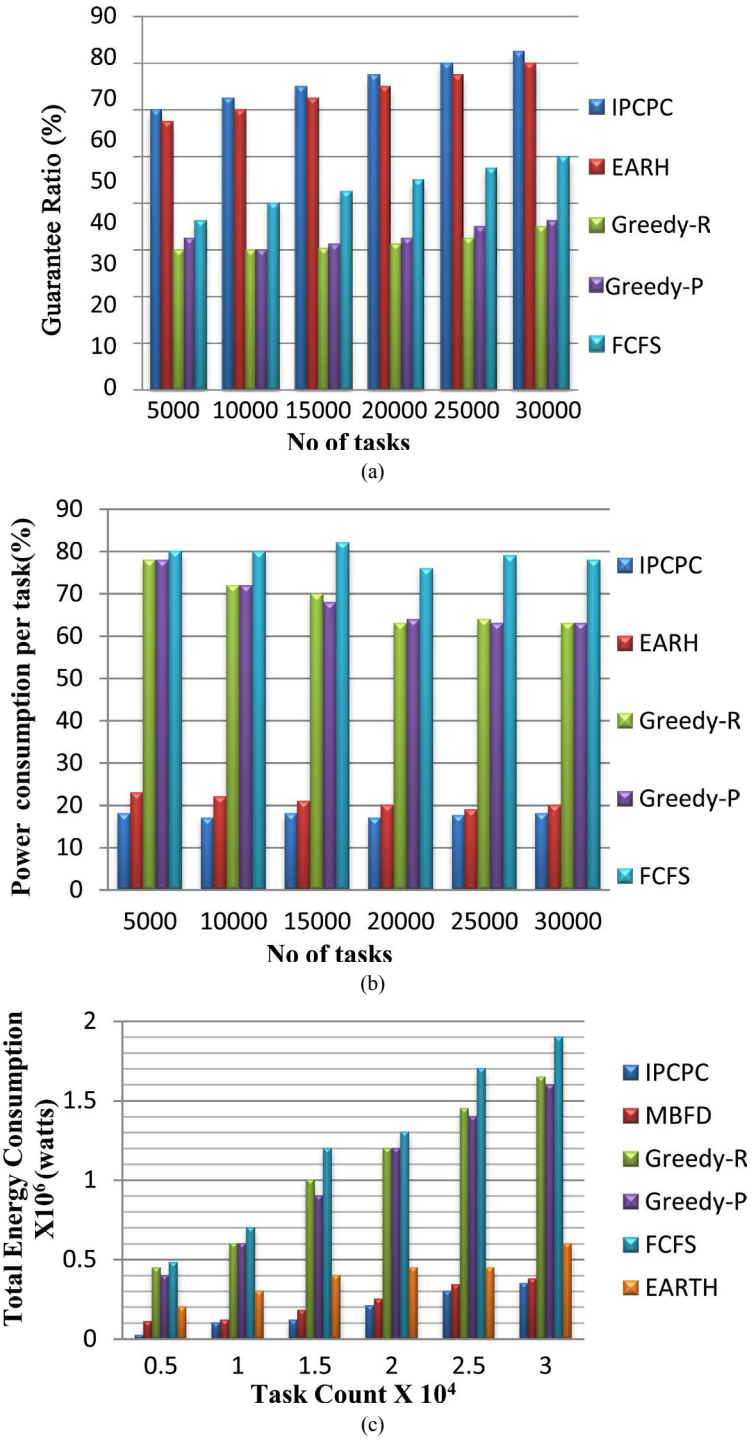


**Figure 6.** (a) Guarantee ratio; (b) power consumption per task to the varying task count; (c) total energy consumption.

**Figure 6(b)** gives the result of power consumption per task to the increasing task count. Considering the above outcomes it is established that proposed IPCPC has least power consumption.

**Figure 6(c)** shows comparison of total energy consumption of tasks. At this juncture six different algorithms are compared. From that it can be verified that the proposed IPCPC achieves more efficient result.

## 4.2. Evaluation Based on Real Data from Google Trace

The above groups of experiments show the performance of the different algorithms in various random inputting tasks. To evaluate the proposed algorithm in practical use, experiments is carried out using data from real world Google trace as input .The details of real world Google trace logs are given in paper [20]. The trace log has information of 29 days. Totally 25 million tasks are recorded in trace log and grouped in 650 thousand jobs are processed in Google in nearly one month. Since there is massive amount of data, only first 5 hours in day 18 [20] were chosen for testing purpose. During these 5 hours 200 thousand tasks were submitted into the cloud. The task counts are varied in time manner. To finish the task it takes 1587 seconds on an average from the submission of task.

The effective utilization of resources for varying task count is shown below in **Figure 7**. The experiments are based on tasks collected from Google trace log.

The total number of tasks guaranteed to meet their deadlines based on tasks collected from Google trace log is shown below in **Figure 8**.

**Figure 9** gives the result of power consumption per task to the increasing task count based on experiments conducted from the tasks collected from Google Trace Log.

**Figure 10** gives the outcome of experiments conducted for total energy consumption for varying task count.

All the above graphs demonstrate the results based on real world trace records. From the analysis of above results it can be proven that the projected framework power management efficient result when compared with previous algorithms.

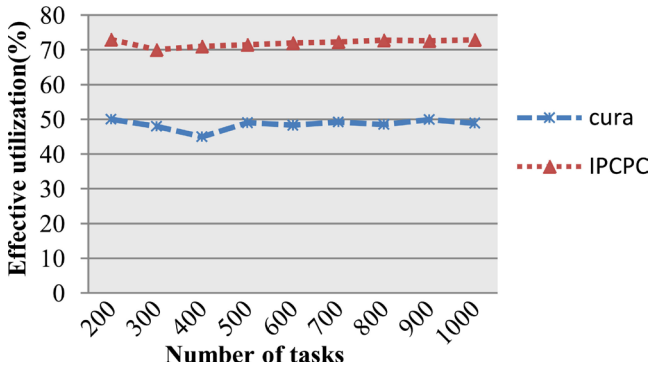


**Figure 7.** Effective utilization of real world Google trace.

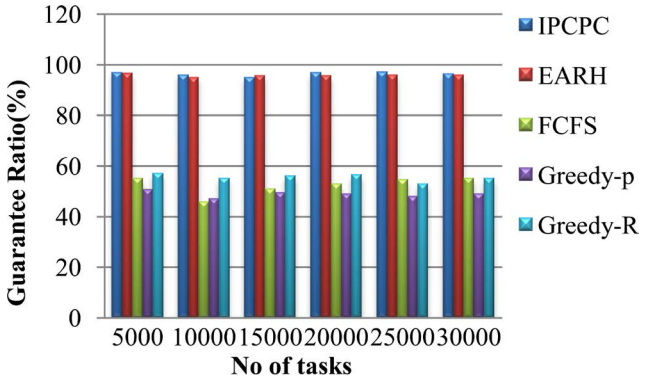


**Figure 8.** Guarantee ratio of real world Google trace.

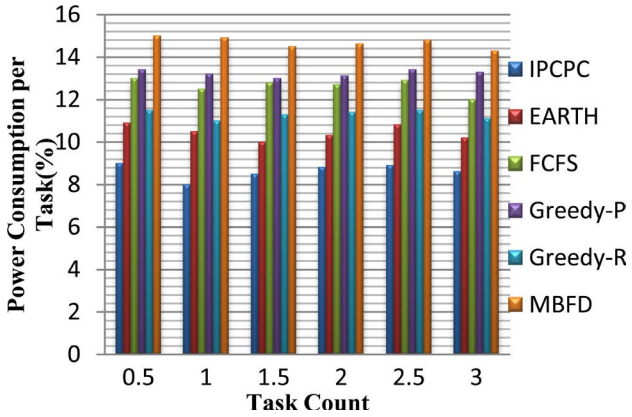


**Figure 9.** Power consumption per task of real world Google trace.

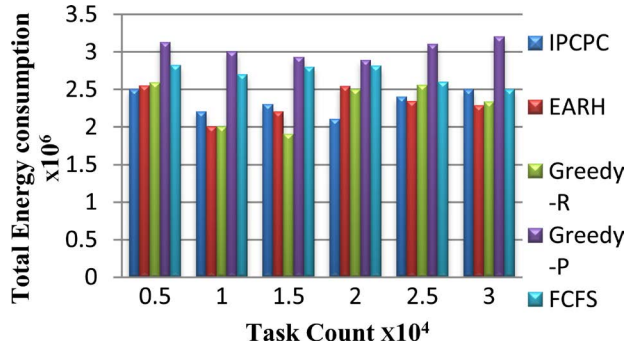


**Figure 10.** Total energy consumption of real world Google trace.

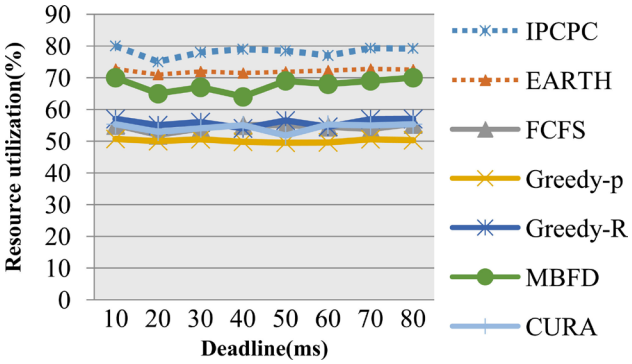


**Figure 11.** Resource utilization of real world Google trace.

## 5. Conclusion

In this paper, the problems of energy conservation in cloud are investigated. As a feasible solution, a framework for power management known as IPCPC is established. It can reduce the overall power consumption and enhances resource utilization. The experimental results prove that IPCPC can efficiently reduce the power consumption than the traditional power aware algorithm. The scrutiny of the experimental results shows that total power consumption per task of server in IPCPC is 9% which proves reduction in the overall power needed.

## References

[1] Mell, P. and Grance, T. (2011) The NIST Definition of Cloud Computing. *National Institute of Standards and Technology Special Publication*, **53**, 1-7.

[2] Banerjee, A., Agrawal, P. and Iyengar, N.Ch.S.N. (2013) Energy Efficiency Model for Cloud Computing. *Internation-*

*al Journal of Energy*, *Information and Communications*, **4**, 29-42. http://dx.doi:10.5121/ijcnc.2012.4. 6.04

[3] Kuribayashi, S. (2012) Reducing Total Power Consumption Method in Cloud Computing Environments. *International journal of Computer Networks & Communications*, **4**, 69-84. http://dx.doi: 10.14257/ijeic.2013.4205

[4] Lu, N., Zhang, M.Y., Li, K.Q., Lob, D.C.-T. and Zhang, Y.Q. (2013) Energy-Efficient Task Scheduling Algorithms on Heterogeneous Computers with Continuous and Discrete Speeds. *Sustainable Computing*: *Informatics and Systems*, **3**, 109-118. http://dx.doi:10.1016/j.suscom.2013.01.002

[5] Liu, L., Wang, H., Liu, X., Jin, X., He, W., Wang, Q. and Chen, Y. (2009) Green Cloud: A New Architecture for Green Data Center. *Proceedings of the* 6*th International Conference Industry Session on Autonomic Computing and Communications Industry Session*, New York, 2009, 29-38. http://dx.doi:10.1145/1555312.1555319

[6] Berral, J.L., Fito, J.O., Julia, F., Nou, R., Guitart, J., Gavalda, R. and Torres, J. (2012) Energy-Efficient and Multifaceted Resource Management for Profit-Driven Virtualized Data Centers. *Future Generation of Computer Systems*, **28**, 718-731. http://dx.doi.org/10.1016/j.future.2011.12.002

[7] Wang, X., Du, Z. and Chen, Y. (2012) An Adaptive Model-Free Resource and Power Management Approach for Multi-Tier Cloud Environments. *The Journal of Systems and Software*, **85**, 1135-1146. http://dx.doi:10.1016/j.jss.2011.12.043

[8] Chase, J.S., Anderson, D.C., Thakar, P.N., Vahdat, A.M. and Doyle, R.P. (2011) Managing Energy and Server Resources in Hosting Centres. *ACM SIGOPS Operating Systems Review*, **35**, 103-116. http://dx.doi:10.1145/502059.502045

[9] Uchechukwu, A., Li, K.Q. and Shen, Y.M. (2012) Improving Cloud Computing Energy Efficiency. *Cloud Computing Congress* (*APCloudCC*) *IEEE Asia Pacific*, **2012**, 53-58. http://dx.doi:10.1109/APCloudCC.2012.6486511

[10] Younge, A.J., von Laszewski, G., Wang, L.Z., Lopez-Alarcon, S. and Carithers, W. (2010) Efficient Resource Management for Cloud Computing Environments. *International Green Computing Conference*, **2010**, 357-364. http://dx.doi:10.1109/GREENCOMP.2010.5598294

[11] Ma, Y., Gong, B., Sugihara, R. and Gupta, R. (2012) Energy-Efficient Deadline Scheduling for Heterogeneous Systems. *Journal of Parallel and Distributed Computing*, **72**, 1725-1740. http://dx.doi:10.1016/j.jpdc.2012.07.006

[12] Wang, L.Z., Khan, S.U., Chen, D., Kołodzie, J., Ranjan, R., Xu, C.-Z. and Zomaya, A. (2013) Energy-Aware Parallel Task Scheduling in a Cluster. *Future Generation Computer Systems*, **29**, 1661-1670. http://dx..doi:10.1016/j.future.2013.02.010

[13] Basmadjian, R., De Meer, H., Lent, R. and Giuliani, G. (2012) Cloud Computing and Its Interest in Saving Energy: The Use Case of a Private Cloud. *Journal of Cloud Computing*, **1**, 1-25. http://dx.doi.org/10.1186/2192-113x-1-5

[14] Baliga, J., Robert, W.A., Ayre, K.H. and Tucker, R.S. (2011) Green Cloud Computing: Balancing Energy in Processing, Storage, and Transport. *Proceedings of the IEEE*, **99**, 149-167. http://dx.doi:10.1109/JPROC.2010.2060451

[15] Talpes, E. and Marculescu, D. (2005) Toward a Multiple Clock/Voltage Island Design Style for Power-Aware Processors. *IEEE Transactions on Very Large Scale Integration Systems*, **13**, 591-603, http://dx.doi:10.1109/TVLSI.2005.844305

[16] Magklis, G., Semeraro, G., Albonesi, D.H., Dropsho, S.G., Dwarkadas, S. and Scott, M.L. (2003) Dynamic Frequency and Voltage Scaling for a Multiple-Clock-Domain Microprocessor. *IEEE Micro*, **23**, 62-68. http://dx.doi:10.1109/MM.2003.1261388

[17] Zhu, X.M., Yang, L.T., Chen, H.K., Wang, J., Yin, S. and Liu, X.C. (2014) Real-Time Tasks Oriented Energy-Aware Scheduling in Virtualized Clouds. *IEEE Transactions on Cloud Computing*, **2**, 168-180. http://dx.doi:10.1109/TCC.2014.2310452

[18] Gutierrez, J.O. and Sim, K.M. (2013) A Family of Heuristics for Agent-Based Elastic Cloud Bag-of-Tasks Concurrent Scheduling. *Future Generation of Computer Systems*, **29**, 1682-1699. http://dx.doi:10.1016/j.future.2012.01.005

[19] Palanisamy, B., Singh, A. and Liu, L. (2014) Cost-Effective Resource Provisioning for Map Reduce in a Cloud. *IEEE Transactions on Parallel and Distributed Systems*, **26**, 1265-1279. http://dx.doi:10.1109/TPDS.2014.2320498.

[20] Moreno, I.S., Garraghan, P., Townend, P. and Xu, J. (2013) An Approach for Characterizing Workloads in Google Cloud to Derive Realistic Resource Utilization Models. *IEEE* 7*th International Symposium on Service-Oriented System Engineering*, Redwood City, 25-28 March 2013, 49-60. http://dx.doi.org/10.1109/sose.2013.24