ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 4, Issue 1, pp: (59-66), Month: January - March 2016, Available at: www.researchpublish.com

Energy Efficient Scheduling Using DVFS Technique in Cloud Datacenters

[1]Amit Kumar Sahoo, [2]Prof. Himanshu Das

[1]M.Tech School of Computer Engineering, KIIT University, Bhubaneswar, 751024, India
[2]Professor at School of Computer Engineering, KIIT University, Bhubaneswar, 751024, India

---

Abstract: **Cloud computing offers utility-oriented IT services to users worldwide. Based on a pay-as-you-go model, it permits hosting of pervasive applications from consumer, scientific, and business domains. However, data centres hosting Cloud applications consume large amounts of energy, causative to high operational-costs and carbon footprints to the atmosphere. Therefore, we'd prefer to introduce an economical Cloud computing solutions which is able to not only minimize operational costs however in addition to minimize the environmental impact. Throughout this paper, we tend to stipulate a field framework by using the DVFS (Dynamic Voltage and Frequency Scaling). During this paper we tried to conduct a survey on different traditional ways of cloud structure and their limitations and at the end of this paper we provide a green energy-efficient scheduling algorithm using the DVFS technique for Cloud computing datacenters.**

Keywords: **Cloud Computing, Energy Efficiency, SLA, QoS, DVFS.**

---

1. INTRODUCTION

A recent report [1] stated: ‘‘Cloud computing, the long-held dream of computing as a utility, has the potential to transform a large part of the IT industry, making software even more attractive as a service’’. Cloud offers significant benefit to IT companies by relieving them from the necessity in setting up basic hardware and software infrastructures, and thus enabling more focus on innovation and creating business value for their services. Moreover, developers with innovative ideas for new Internet services no longer require large capital outlays in hardware to deploy their service or human expenses to operate it [1]. Until recently, high performance has been the sole concern in data center deployments, and this demand has been fulfilled without paying much attention to energy consumption. According to a report on *“Revolutionizing Data Center Energy Efficiency”*, A typical data center consumes as much energy as 25,000 households. Therefore, we need Energy efficient Cloud computing solutions that can not only minimize operational costs but also reduce the environmental impact. In this Thesis work we compare some prevoiusly used algorithms and tried to propose a new scheduling algorithm, that can efficiently increase resource utilization; hence, it can decrease the energy consumption for executing jobs. The performance of executing jobs is not sacrificed in our scheme. We provide a green energy efficient scheduling algorithm using the DVFS technique for Cloud computing datacenters.

2. RELATED WORKS

One of the primary works, during which power management has been applied at the data centre level, in [2]. During this work the authors have planned a way for minimisation of power consumption during a heterogeneous group of computing nodes serving multiple web-applications. The most technique applied to reduce power consumption is concentrating the work to the minimum of physical nodes and switch idle nodes off. This approach needs handling the power / performance trade-off, as performance of applications is degraded because of the workload consolidation. Needs to the output and execution time of applications are outlined in SLAs to confirm reliable QoS. The proposed algorithmic program periodically monitors the load of resources (CPU, disk storage and network interface) and makes choices on switching nodes on / off to reduce the power consumption, whereas providing the expected performance. The original load balancing isn't handled by the system and should be managed by the applications. The algorithm runs on a master node that creates a single point of Failure (SPF) and should become a performance bottleneck during a large system. Additionally, the authors have seen that the reconfiguration operations are time consuming, and therefore the algorithm adds or removes just one node at a time, which can even be a reason for slow reaction in large-scale environments. The proposed approach is applied to multi-application mixed-workload environments with fixed SLAs.

In [3] the thought of the matter of energy efficient management of uniform resources in net hosting centres. The most challenge is to see the resource demand of every application at its current request load level and to allot resources within the most effective method. To affect this downside, the authors have applied associate economic framework: services \bid" for resources in terms of volume and quality. This permits negotiation of the SLAs per the available budget and current QoS needs, i.e. balancing the value of resource usage (energy cost) and therefore the benefit gained because of the usage of this resource. The system maintains an active set of servers selected to serve requests for every service. The network switches are dynamically reconfigured to vary the active set of servers when necessary. Energy consumption is reduced by change idle servers to power saving states (e.g. sleep, hibernation). The system is targeted at the online workload, that results in a \noise" within the load data. The authors have self-addressed this downside by applying the applied mathematics “flipip-op” filter, that reduces the amount of unproductive reallocations and results in a lot of stable and economical control. The proposed approach is appropriate for multi-application environments with variable SLAs and has created a foundation for varied studies on power efficient resource allocation at data centre level. However, the system deals only with the management of the central processing unit, however doesn't consider different system resources. The latency because of change nodes on /off is also not taken into consideration. The authors have noted that the management rule is quick once the workload is stable, however seems to be comparatively dear throughout significant changes within the work. Moreover, likewise [2], various software configurations aren't handled, which may be addressed by applying the virtualization technology.

In [4] have investigated the problem of power efficient resource management during some single web-application surroundings with fixed SLAs (response time) and load balancing handled by the application. As in [3], 2 power saving techniques are applied: switch power of computing nodes on / off and Dynamic Voltage and Frequency Scaling (DVFS). The most idea of the policy is to estimate the overall C.P.U. frequency needed to produce the mandatory latency, confirm the optimum range of physical nodes and set the proportional frequency to all or any the nodes. However, the transition time for change the energy of a node isn't thought of. Solely a single application is assumed to be run within the system and, like in [2], the load balancing is meant to be handled by an external system. The algorithm is centralized that makes an SPF and reduces the measurability. Despite the variable nature of the work, unlike [3], the resource usage data aren't approximated, which ends in doubtless inefficient selections because of fluctuations.

In [5] have planned cooperation between ISPs and content suppliers that enables the accomplishment of an economical coincident allocation of reckon resources and network methods that minimizes energy consumption underneath performance constraints. Koseoglu and Karasan have applied an analogous approach of joint allocation of machine resources and network methods to Grid environments supported the optical burst change technology with the target of reduction of job completion times. Tomas et al. have investigated the matter of scheduling Message Passing Interface (MPI) jobs in Grids considering network data transfers satisfying the QoS needs. Dodonov and de Mello have planned an approach to planning distributed applications in Grids supported predictions of communication events. They need proposed the migration of act processes if the migration price is under the value of the anticipated communication with the target of minimizing the overall execution time. They need shown that the approach is effectively applied in Grids; but, it's not viable for virtualized data centers, because the VM migration price is over the method migration price. Gyarmati and Trinh have investigated the energy consumption implications of knowledge centers' network architectures. However, optimisation of network architectures is applied solely at the data center style time and can't be applied dynamically.

The dynamic voltage and frequency scaling (DVFS) technique is commonly used to reduce the power consumption of electrical devices such as cell phones, PDAs, and PCs. The power consumption of an integrated circuit is proportional to the simple formula ‘fcv^2’, with f the frequency, c the capacitance, and v the voltage. Thus, the supply voltage and work frequency profoundly affect the energy consumption of integrated circuits. The DVFS enables integrated circuits to run at different combinations of frequencies and voltages. Voltage supply can be increased or decreased depending upon circumstances. The DVFS can dynamically lower down the supply voltage and work frequency to reduce the energy consumption while the performance can satisfy the requirement of a job.

In [6], Author introduced an interval number theory to describe the uncertainty of the computing environment and a scheduling architecture to mitigate the impact of uncertainty on the task scheduling quality for a cloud data center. Based on this architecture, they present a novel scheduling algorithm (PRS1) that dynamically exploits proactive and reactive scheduling methods, for scheduling real-time, aperiodic, independent tasks. To improve energy efficiency, we propose three strategies to scale up and down the system's computing resources according to workload to improve resource utilization and to reduce energy consumption for the cloud data center.

In [7], an energy-aware resource allocation heuristic for efficient management of datacentres for Cloud computing is presented. The proposed method can improve energy efficiency of the data centre, while delivering the negotiated Quality of Service (QoS). This work provides architectural principles for energy-efficient management of clouds, energy-efficient resource allocation policies and scheduling algorithms. Their method has to sacrifice system performance.

3. OBJECTIVE

In these above models we can see that the author tried to solve the major problems (like SLA, Performance, Energy efficiency, Power consumption, Heating problem etc.) in the Cloud computing. They focused on a single problem on their work, by ignoring the other major problems. Because all the constraints are very important in this widely used computing model, so we cannot ignore any of them. We have to short out the issue of energy consumption as well as we cannot violet the Service Level Agreement. Unlike the other scheduling algorithm here in this Thesis work we tried to solve the major problems like Energy consumption, heating issue in host machines, and improving SLA without any degradation in performance. For reducing energy consumption, we separate the different type of task by considering different time constraints so that they can have different priority level for scheduling. Every server will regulate its work frequency as per its workloads. To standardize capacities of heterogeneous servers, we use DVFS technique. The DVFS enables integrated circuits to run at different combinations of frequencies and voltages. Voltage supply can be increased or decreased depending upon circumstances.

4. SYSTEM ARCHITECTURE

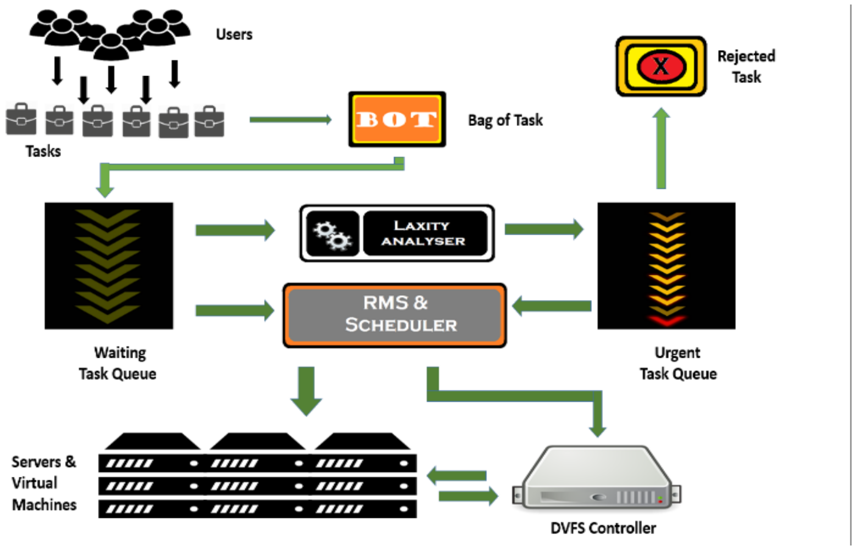


Figure: 2 System Architecture

5. SYSTEM OVERVIEW

At first user have to submit the task along with all the related information to its job or bag of task (BoT) like maximum and minimum operating frequency, deadline and information about resource. We consider a different approach, which aims at reducing the power required to execute urgent, CPU-intensive Bag of Tasks (BoT) applications on cloud infrastructures.

Service Level Agreement (SLA) is a very important contract between the service provider and consumer. The allocation algorithm has to obey the SLA to assign resources for the job given by the consumer. In our work, we also provide SLA levels for consumers to select. An example is as follows.

Level	Numbers of VMs
0	1-200
1	201-400
2	401-600
3	701-800
4	801-1000

An example of SLA levels and related numbers of VMs

The minimum and maximum working frequency requirements for executing job a are 100 and 600 MHz, respectively. Three VMs in a cloud are available to deploy this job. The frequency specifications of these three VMs are listed as follows:

VM V1 has Fmin = 100 MHz and Fmax = 500 MHz

VM V2 has Fmin = 100 MHz and Fmax = 600 MHz

VM V3 has Fmin = 200 MHz and Fmax = 800 MHz

An urgent application is a High Performance Computing application whose execution needs to complete before a user-defined deadline because of its utilization in sensitive contexts, such as disaster management and healthcare. We define a deadline-constrained application as an application that needs to have its execution completed before a user-defined soft deadline. Notice that, in contrast to a hard deadline, a soft deadline does not render the computation useless if the deadline is violated. Instead, there exists a utility value associated to the computation whose value is maximized if the application completes by the deadline, and is increasingly reduced as the completion is delayed.

Our energy model is derived from the power consumption model in complementary metal-oxide semiconductor (CMOS) logic circuits. The power consumption of a CMOS-based microprocessor is defined to be the summation of capacitive, short-circuit and leakage power.As we know that:

$$\mathrm{Wi} = \mathrm{Pi} * \mathrm{Ri} \qquad \ldots\ldots(1)$$

Where P is the unit power cost of server/VM i and Ri is resources used by server/VM. Wi can aid the allocation algorithm to decide which server/VM is assigned to which job. It acts as a weight and priority in the algorithm.The capacitive power (dynamic power dissipation) is the most significant factor of the power consumption. The capacitive power (P_c) is defined as

$$\mathrm{Pc} = \mathrm{AC}V^2\mathrm{f} \qquad \ldots\ldots(2)$$

Where A is the number of switches per clock cycle, C is the total capacitance load, V is the supply voltage, and f is the frequency. Equation 3 clearly indicates that the supply voltage is the dominant factor; therefore, its reduction would be most influential to lower power consumption. The energy consumption of the execution of a precedence-constrained parallel application used in this study is defined as:

$$E = \sum_{i=0}^{n} ACV_i^2 f.w_i^* \qquad \ldots\ldots(3)$$

$$= \sum_{i=0}^{n} \propto V_i^2 w_i^*$$

Where V_i is the supply voltage of the processor on which task ni executed, and wi* is the computation cost of task ni (the amount of time taken for n's execution) on the scheduled processor.

Then the task are waiting for execution in a single queue call as Waiting Queue. The task are regularly checked by their laxity by the help of the Laxity Analyser.

$$L_i = d_i - \frac{l_i}{\min\{c_{ik}\}} - ct \qquad \text{......................(4)}$$

Where l_i and d_i are the computation length upper bound and deadline of task t_i, respectively; $\min\{c_{jk}\}$ is the computing capacity lower Bound of the VM with minimal CPU performance; $l_i/(\min\{c_{jk}\})$ represents the upper bound of task t's maximal execution time and ct is the current time.

If the laxity value exceeds the pre-determined value then the task directly goes to Urgent Task Queue (UQ), if don't then stays in the waiting task queue.

The Resource Management System (RMS) monitors all the resource related to each task and according to availability the scheduler schedules the tasks. The scheduling algorithm works on minimum frequency and maximum frequency of the task and virtual machines. The requirements of a job are given in terms of maximum and minimum frequencies. The minimum frequency is to ensure the lowest limit of the executing performance for a job. A list of available servers is given by the RMS. The RMS has to list the server statuses including used resources and remainder capacities. Servers chosen for a job have to satisfy two inequalities as follows:

$S_i\,(F_{min}) >= Job\,(F_{min})$ …………………………………. (a)

$S_i\,(F_{max}) <= Job\,(F_{max}) <= S_i\,(F_{max}) + \alpha$ ………………. (b)

The first inequality indicates that the minimum frequency F Max of the chosen server has to be bigger than or equal to the minimum frequency requirement of the job. This inequality guarantees the executing performance of the job. Inequality 2 shows that the maximum frequency F of the chosen server has to be smaller than or equal to F max max of the job. From multiple severs whose maximum frequencies are smaller than F of the job, we choose the server whose maximum frequency is the closest to F max of the job. This purpose is to ensure that the job cannot overuse resources. In this constraint, the energy consumption of the job is also reduced.

Process of scheduling algorithm:-

Step 1. Receives a job and its maximum frequency, minimum frequency, deadline and also SLA level from the user.

Step 2. All task deposited into WTQ (Waiting Task Queue)

Step 3. Calculate Laxity value each job by Laxity analyser

Step 4. If laxity value greater than the predefined value it goes to the UTQ (Urgent Task Queue)

Step 5. Scheduler Schedules the job by checking 2 inequality constraint and and arrange them WT and UT with higher priority.

Step 6. RMS monitors the VMs and servers and sends all the information to DVFS controller.

Step 7. DVFS controller applies RDVFS algorithm and decides appropriate frequency so that processor can operate efficiently.

Step 8. If the task cant not be completed within its deadline it puts the task into reject list.

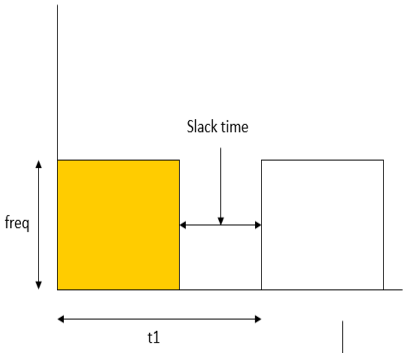
The values of required frequency for each server are computed in the first process of our scheme; the frequency (f) to be given of each server is determined. To minimize the power consumption, the DVFS controller has to supply applicable amount of frequencies for each server by adjusting to the preferred gear. The DVFS controller sets the appropriate gear to supply the proper frequency and voltage to each server. DVFS technique and task scheduling can be combined in two ways:

(1) Schedule generation

(2) Slack time reclamation

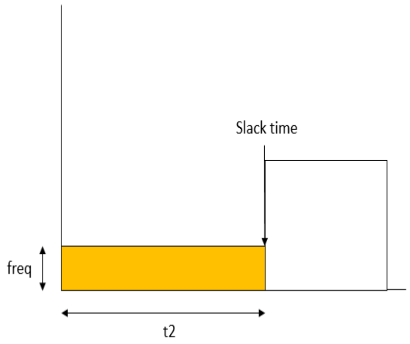
In schedule generation, tasks graph are (re)scheduled on DVFS-enabled processors in a global cost function including both energy saving and makespan to meet both energy and time constraints at the same time. In slack time reclamation, which works as a post-processing procedure on the output of scheduling algorithms, the DVFS technique is used to minimise the energy consumption of tasks in a schedule generated by a separate scheduler. In our model we focus on slack time reclamation by using RDVFS (Relational DVFS) technique.

There are a lot of different techniques to implement DVFS differently to optimise the energy consumption like MET DVFS, Min-Min DVFS, Max-Min DVFS, OLB DVFS, fast-greedy DVFS, Optimum continues DVFS and Relational DVFS etc. Here we are going to discuss difference between task scheduling without DVFS and Optimum continues DVFS and R-DVFS.



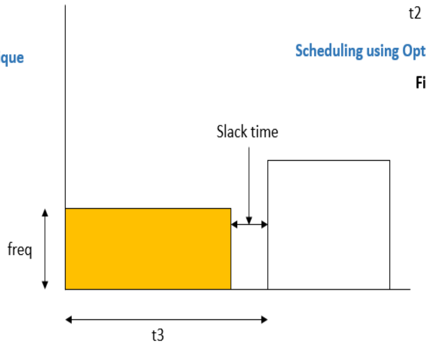
Scheduling using traditional DVFS technique

Fig 1(a)



Scheduling using Optimum continues DVFS technique

Fig 1(b)



Scheduling using Relational DVFS technique

Fig 1(c)

Traditional DVFS:

DVFS enabled processors can execute a task by using a discrete set of voltage-frequency pairs, (f_i, v_i), in which $\{v_1 < v_2 < v_3 . . . < v_n\}$ and $\{f_1 (\min) < f_2 < . . . < f_n (\max)\}$ The whole power consumption (P) is estimated as [14]:

$$P_d = \lambda f v^2$$

$$f \propto \frac{(v-v_t)^2}{v_t} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(5)$$

Where f, λ and v represent the processor's working frequency, the effective capacitance, and the processor's working voltage, respectively. Note that is a threshold voltage usually provided by a manufacturer. The general relationship between voltage, frequency and power is:

$$If\ (f_i, v_i) < (f_j, v_j) => P\ (f_i, v_i) < P\ (f_j, v_j)$$

The overall energy consumption of k_{th} *task* ($A^{(k)}$) in DAG is calculated as :

$$E\ (k)\ =\ P_d t_i^{(k)} + P_{idle}(T^{(k)} - t_i^{(k)}) \qquad \text{........(6)}$$

Where P is the energy a processor consumes when it is in idle. With the almost always true assumption that is constant, and v >> v_t, the relationship between frequency and voltage becomes proportional, i.e. $f\ \alpha\ v$; therefore, Eqn. 5 is simplified as:

$$\mathrm{Pd} = \lambda f^3 + \Upsilon \qquad \text{..................................(7)}$$

Optimum Continuous Frequency:

The optimal approach to remove slack time and, as a result, reduce the energy consumption of a processor, is for the processor to perform a task using a continuous frequency (Figure 2-b).

Theorem: If processor frequency is continues (unrealistic assumption), the optimum energy for *k*th task is obtained when the task covers the whole task's slack time T(k).

Proof: the result in theorem 1 shows that when a frequency covers the whole slack time it gives the optimum power consumption. Note that this frequency may not exist unless the frequency set is continuous. Referring to theorem, for k_{th} task ($A^{(k)}$), the optimum continuous frequency and its related energy are defined as $f_{opt\text{-}cont.}$ and $E_{opt\text{-}cont.}$ and are calculated as:

$$f^{(k)}_{opt-cont} = f_N \frac{t^{(k)}_{os}}{T^{(k)}} \qquad \text{...(8)}$$

$$E^{(k)}_{opt-cont.} = \left(\propto \left(f^{(k)}_{opt-cont.}\right)^3 + \gamma\right) T^{(k)}$$

In actual systems, however, frequencies must be chosen from a discrete set of frequencies. Also, finishing a task by its deadline may require choosing a frequency that is faster than the optimal frequency. Therefore, the optimal discrete frequency of *k*th task is the first frequency in the discrete set larger than $f^{(k)}_{opt-cont.}$. this discrete frequency and its associated time are $f^{(k)}_{RD}$ and $t^{(k)}_{RD}$, respectively. The algorithm calculating this frequency is referred to as RDVFS for our comparison.

Reference Dynamic Voltage-Frequency Scaling (RDVFS):

RDVFS is a simplified version of the algorithm introduced by Kimura et al in [12] for power-scalable high performance clusters supporting DVFS. It reduces the energy consumption of processors by selecting the smallest available processor frequency (f_{RDVFS}) capable of finishing a task in a given time frame (Figure 2-c). The details of RDVFS algorithm are shown in below Figure.

For each task assigned to a processor, $f^{(k)}_{RDVFS}$ which is the first frequency larger than optimal frequency ($f^{(k)}_{opt-cont.}$) calculated from Eqn. 3-5, is likely to be the best discrete frequency candidate to execute the task within the given time frame and covering its related slack time. As mentioned before, a major limitation of the RDVFS technique is the usage of only one frequency to execute the task.

6. CONCLUSION AND FUTURE WORK

In this Thesis work, we described different approaches and present a green energy-efficient scheduling algorithm which makes the use of Laxity Analyser for successfully execute each and every job individually so that The VMs are selected according SLA level given by user. Our method can satisfy the minimum resource requirement of a job and prevent the excess use of resources by giving. Hence, we can increase the resource utilization. The RDVFS technique is used to control the supply voltage and frequency for servers in Cloud computing. This technique can reduce the energy consumption of a server when it is in the idle mode or the light workload.

Though we have not implement this approach, so we cannot say it will give best result. In future we want to implement it in software level and try to compare with existing approaches to produce a statistical result. If it will show good result, we will try to implement it in real life also. If it need we will also adopt new technologies.

REFERENCES

[1] Armbrust, Michael, et al. "A view of cloud computing." *Communications of the ACM* 53.4 (2010). pp. 50-58.

[2] Pinheiro, Eduardo, et al. "Load balancing and unbalancing for power and performance in cluster-based systems." *Workshop on compilers and operating systems for low power*. Vol. 180. 2001. pp. 182-195.

[3] J. Chase, Jeffrey S., et al. "Managing energy and server resources in hosting centers." *ACM SIGOPS Operating Systems Review*. Vol. 35. No. 5. ACM, 2001. pp. 103-116.

[4] Elnozahy, EN Mootaz, Michael Kistler, and Ramakrishnan Rajamony. "Energy-efficient server clusters." *Power-Aware Computer Systems*. Springer Berlin Heidelberg, 2003. pp. 179-197.

[5] Chiaraviglio, Luca, and Ibrahim Matta. "Greencoop: cooperative green routing with energy-efficient servers." *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*. ACM, 2010. pp. 191-194.

[6] Chen, Huangke, et al. "Towards energy-efficient scheduling for real-time tasks under uncertain cloud computing environment." *Journal of Systems and Software* 99 (2015). pp. 20-35.

[7] Beloglazov, Anton, Jemal Abawajy, and Rajkumar Buyya. "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing." *Future generation computer systems* 28.5 (2012). pp. 755-768.

[8] Kaur, Rajleen, and Amanpreet Kaur. "*A Review Paper on Evolution of Cloud Computing, its Approaches and Comparison with Grid Computin*g." (2014). pp. 262-271.

[9] Yao, Frances, Alan Demers, and Scott Shenker. "A scheduling model for reduced CPU energy." In *Foundations of Computer Science, 1995. Proceedings. 36th Annual Symposium on*, pp. 374-382. IEEE, 1995.

[10] Ishihara, Tohru, and Hiroto Yasuura. "Voltage scheduling problem for dynamically variable voltage processors." In *Low Power Electronics and Design, 1998. Proceedings. 1998 International Symposium on*, pp. 197-202. IEEE, 1998.

[11] Cong, Jason, and Karthik Gururaj. "Energy efficient multiprocessor task scheduling under input-dependent variation." In *Proceedings of the Conference on Design, Automation and Test in Europe*, pp. 411-416. European Design and Automation Association, 2009.

[12] H. Kimura, M. Sato, Y. Hotta, T. Boku, and D. Takahashi, "Emprical study on Reducing Energy of Parallel Programs using Slack Reclamation by DVFS in a Powerscalable High Performance Cluster," in *Cluster Computing, 2006 IEEE International Conference on*, 2006, pp. 1-10.

[13] Mochocki, Bren C., Xiaobo Sharon Hu, and Gang Quan. "A unified approach to variable voltage scheduling for nonideal DVS processors." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 23, no. 9 (2004): 1370-1377.

[14] Oh, S.-H., Yang, S.-M, 1998. A modified least-laxity-first scheduling algorithm forreal-time tasks. In: IEEE 5th International Conference on Real-Time Computing Systems and Applications. IEEE, pp. 31–36.