Journal of Engineering and Applied Sciences 14 (5): 1699-1705, 2019
ISSN: 1816-949X
© Medwell Journals, 2019

Performance Evaluation of Proposed Algorithm in Real-Time Streaming Warehouses

[1]D.S. Misbha and [2]J. R. Jeba
[1]Department of Computer Applications, Nesamony Memorial Christian College, Tamil Nadu, India
[2]Department of Computer Applications, Noorul Islam Centre for Higher Education,
Tamil Nadu, India

Abstract: The need to provide up-to-date information, streaming warehouses are used to screen multipart systems such as web site complexes, data centers and world-wide networks, congregating and comparing encumbered collections of happenings and measurements. For profound analysis and for rapid responses, both chronological data and concurrent data that raising the problems in streaming warehouses were used. The data warehouse gathers a large number of streaming data provisions that are generated by external sources and reach target asynchronously. Scheduling updates is the most important processes that are concerned severely in streaming warehouses. Scheduling algorithms provided for loading data in concurrent data warehouses that are used in the applications such as online monetary trading, IP network screening and credit card scam detection. In this study, the evaluation of performance analysis of RECSS algorithm is described. The first objective of this study is to schedule the updates one by one on one or more processors in a way to reduce the total mustiness. In turn to verify the total fairness, the second objective of this study is to limit the maximum "extend" in which to define (approximately) as the ratio between the periods of time of an update waits till being the process is concluded and the length of each updates. When compared to the previous research, it deserves the mustiness is provided that the processors are adequately fast and find that only those update propagation algorithms which enforce no scheduling constraints are tolerable for use in a concurrent streaming warehouse.

Key words: RECSS, CS, EARH, NMEARH, rolling horizon, virtualized clouds

INTRODUCTION

Power-related costs have become a major cost-effective aspect for IT infrastructures and data-centers because of the power's price amplification. Companies are at present focusing on the need to improve power efficiency. Cloud applications are considered in remote Data Centers (DCs) where more power servers and storage systems are placed. A rapid development of claim for cloud based services fallout into organization of enormous data centers overwhelming high amount of electrical power. Power efficient model is essential for entire infrastructure to minimize functional costs while sustaining essential Quality of Service (QoS). Power optimization can be achieved by mingling resources as per the present utilization and providing well-organized virtual network topologies.

Chiesi *et al*. (2015) a power-aware job scheduling algorithm was presented based on efficient allocation of the computing workload to the resources on heterogeneous CPU-GPU architectures. The algorithm built by Hsu and Feng (2005) presents a power-aware DVFS run-time system that presents energy reduction with minute performance loss. Jeba amd Victor (2011a) proposed an efficient SMine (Sorted Mine) algorithm for discovering repeated tasks and to decrease the number of jobs in the list. In SMine (Jeba and Victor, 2011b) algorithm, the set of all repeated itemsets are extracted in the database by limiting the number of scans. Petrucci *et al.* (2009) put forward a dynamic configuration approach for energy optimization in virtualized server clusters and sketches an algorithm to dynamically manage it. Goiri *et al*. (2010) presented a framework that gives a consolidation methodology using turning on/off machines, machine learning techniques and power-aware consolidation algorithms to compact with hesitant information while maximizing performance. Wang *et al*. (2012) a method was proposed to vigorously control the peak power while maintaining system performance as high as possible. According to latest studies, the average Resource Utilization (RU) in most of the data centers is <30% (Barroso *et al*.,

Corresponding Author: D.S. Misbha, Department of Computer Applications, Nesamony Memorial Christian College, Tamil Nadu, India

2013) and the power consumption of redundant resources is >70% of peak power (Fan *et al*., 2007).

Problem description: In this study, a virtualized cloud is embattled which is demonstrated by an innumerable set of hosts, H = $\{h_1, h_2, , \ldots,\}$ which supplies the hardware infrastructure for creating virtualized resources to persuade the user's requirements. The self-motivated host is modeled with n number of factors. For a given host h_k, it is differentiated by its resource routine defined by Million Instructions Per Second (MIPS) (Wu *et al*., 2012; Calheiros *et al*., 2011) with the quantity of RAM and the network bandwidth i.e., $h_k = \{c_k, r_k, n_k\}$ where the c_k, r_k and n_k represents the CPU ability, RAM and for the network bandwidth of the kth host, respectively. Each host holds a set of Virtual Machines (VMs) i.e, $V_k = \{\upsilon\tilde{o}_{1k}, \upsilon_{2k}, \ldots, \upsilon_{kk}\}$. For each Virtual Machine (VM) υ_{jk}; c (υ_{jk}), r (υ_{jk}) and n (υ_{jk}) are used to represent the fractions of CPU performance, network bandwidth and amount of RAM. Based on the workload, the virtual machines VMs are dynamically established and bring to a halt on a single host.

MATERIALS AND METHODS

Overview of NMEARH, EARH and Cuckoo search

NMEARH: NMEARH utilizes the rolling-horizon optimization policy that is able to make jobs with firm deadlines finish earlier, so, the schedulability is drastically developed. NMEARH does not employ the Virtual Machine (VM) migration while allocating concurrent jobs. In addition, NMEARH was lacking in providing fine trade-off between guarantee ratio and entire energy consumption.

EARH: EARH system implements the rolling-horizon optimization to efficiently promise the schedulability of concurrent tasks and at the same time motivating to save energy by active VMs consolidation. It provides the basic information about the virtualization technology in cloud computing and how it acts in the cloud computing surroundings. The method to use a rolling horizon approach on supply string optimization and scheduling crisis has been applied in dissimilar contexts. Although, a rolling horizon is generally used when uncertainties in data subsist, a rolling horizon is also applicable to reduce the size of the problem. Both a forward and backward rolling horizon method is used for scheduling multipurpose tasks. A rolling horizon approach is used to optimize a hierarchical development system. EARH can proficiently improve the scheduling quality of others in different workloads and is appropriate for energy aware scheduling in virtualized clouds.

Cuckoo search: CSA algorithm is based on the necessitate litter parasitic activities of some Cuckoo species in permutation with the Levy flight activities of some birds and fruit flies. CSA is a new meta-heuristic approach that models the ordinary activities of Cuckoos. The idea behind this algorithm is that each Cuckoo lays one egg at a time and deposits it in an arbitrarily selected nest. The best nests with high quality eggs (solutions) will be carried out over to the next generations. Likewise, the best VM is selected for executing the newly coming tasks.

Proposed RECSS strategy: A new algorithm called the RECSS (Rolling Energy Cuckoo Scale Scheduling) algorithm is proposed to maintain the job scheduling without any deadlines as well as to keep energy saving. The overall architecture of the proposed RECSS strategy is shown Fig. 1.

The proposed RECSS strategy involves a Data mining process, job scheduling process and job execution process which was shown in Fig. 1. Relevant datasets are extracted from the data warehouse during data mining process. Job scheduling and resource allocation process takes place in job scheduling phase. In the job execution phase, the scheduled jobs are executed within their deadlines. Resource utilization is analysed in job scheduling phase and execution time and power consumption were analysed during job execution.

Rolling-horizon optimization: The proposed approach locates the entire new and waiting task into a rolling-horizon and then they are scheduled by the Cuckoo scale scheduler. The jobs that scheduled are permitted to be adjusted for the system schedulability and possibly low power consumption. The pseudocode for rolling-horizon optimization is given below Algorithm 1.

Algorithm 1; Pseudocode for rolling-horizon optimization:

```
1:   for each new task $t_i$
2:              Q←NULL; R←NULL; ECS←NULL
3:   Task are added into Q and Memory Allocation in VM
4:              Add a new task $t_i$ into set Q
5:                     for ($t_i$) each task $t_n$ do
6:              if $t_i>a_i$ then
7:                     Add task $t_i$ into set Q
8:                     Add task $t_a$ into VM
9:              end if
10:             else if $t_i<a_i$ then
11:                    $rt = rt+t_{w;}$
12:                    VM←rt
13:                    Update the ready time of Task
14:                    Update the Waiting Task in to VM
15:             end if
```

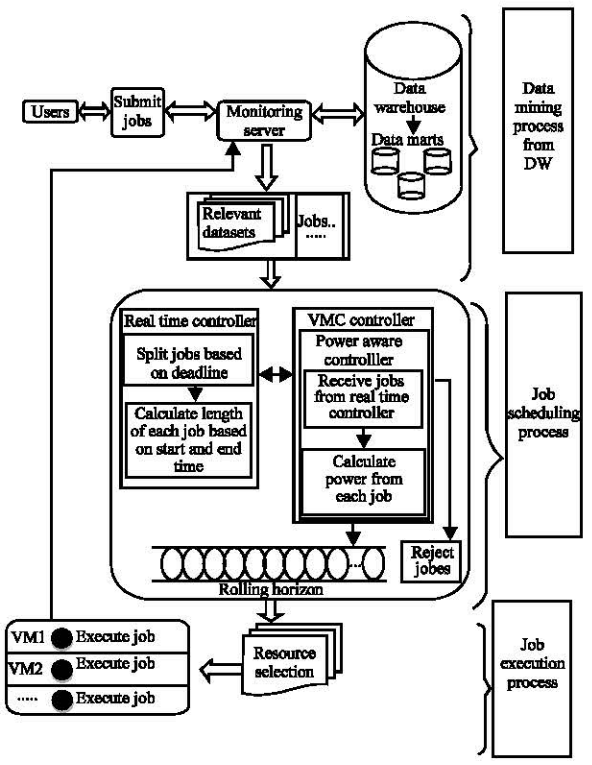

Fig. 1: Overall architecture

In the pseudocode of rolling-horizon optimization, before adding the innovative tasks into the rolling-horizon Q, it is set to null (line 2). When the new task enters it adds into the set Q and all the waiting tasks are also added into the set Q (lines 4-7). The ready time of each VM is computed and this ready time is the starting time of the task in the VM (lines 10-12). Bring up to date the ready time of the each VM and update the waiting task into the VM (lines 13 and 14). After that the tasks are scheduled.

Energy-aware task scheduling and execution: After updating all the tasks into the VM, the tasks are scheduled using Cuckoo scale scheduler based on the deadline. The tasks are then assigned priority. The priority of tasks may be high, medium and low. Based on the priority, the jobs are executed, i.e., the jobs with higher priority are executed first. The pseudocode of energy-aware task scheduling and execution algorithm is given in algorithm 2.

Algorithm 2; Pseudocode of RECSS algorithm:

```
1:  After updating each tasks
2:  Sort tasks in Q by their deadlines in a non-descending order
3:          for each task $t_i$ in set Q do
4:          Schedule task $t_i$ by ENERGY Cuckoo Scheduling Algorithms;
5:          if calendar(ti) ! = 1 then
6:                  Reject task $t_q$
7:          end if
8:          else
9:  Update Scheduling decisions;
10:         ECS = ECS+$t_i$
11:         end else
12:         end for
13: Execute the Schedule Task from the VM
14:         for each task $t_i$ in set $Q_n$ do
15:                 Start while
16:                 while($t_i$ < Qn )
17:                 SE = ECS($t_i$)
18:                 i++;
19:         End while
```

The energy-aware Cuckoo scale scheduling algorithm is in heuristic fashion. It allocates each task to a VM in a way to aggressively meet task's deadlines while conserving energy consumption. After updating each task into the set Q, the tasks are sorted based on their deadline (lines 1-3). After sorting, the tasks are scheduled using energy-aware Cuckoo scale scheduling algorithm. If the VM can meet the tasks deadline, it accepts the task, otherwise rejects it (lines 4-6). After that the scheduling decisions are updated and the energy consumption rate is calculated for each tasks. The scheduled tasks are then executed (lines 9-13).

RESULTS AND DISCUSSION

Performance evaluation: To demonstrate the performance improvements gained by RECSS, three algorithms are compared-NMEARH (Non-Migration EARH), EARH (Energy-Aware Rolling-Horizon) algorithm and CS (Cuckoo Search) algorithm. The performance metrics are compared in terms of task count. The task count varies in the range 100.

Comparison of recss with nmearh, earh and cs algorithm for resource allocation: The proposed algorithm is compared with NMEARH, EARH and CS. The performance metrics such as resource allocation, energy consumption and resource utilization are considered for performance comparison. Resource allocation is the total resources allocated to each VMs. Energy consumption is the total energy consumed be each host for executing the task. Resource utilization is the average host utilization.

The resources allocated to each VM must be low and thus it reduces the cost. If the allocation of resources is maximum, it will costs high. Figure 2 shows it is observed that the proposed RECSS algorithm allocates only a minimum amount of resources than the existing NMEARH, EARH and CS algorithms. For RECSS, the percentage of resource allocation increases slightly when the number of task increases. But in NMEARH, EARH and CS, the percentage of resource allocation increases

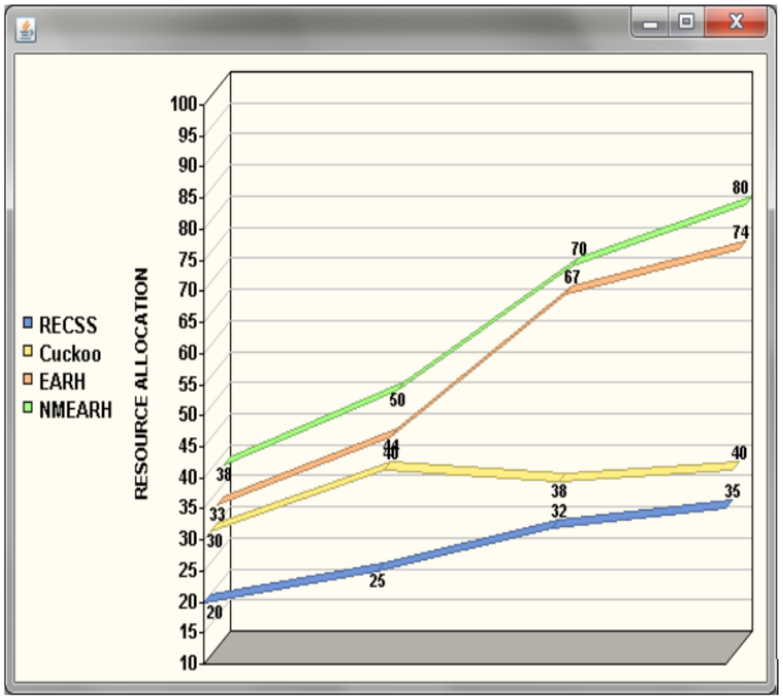

Fig. 2: Task count vs. resource allocation

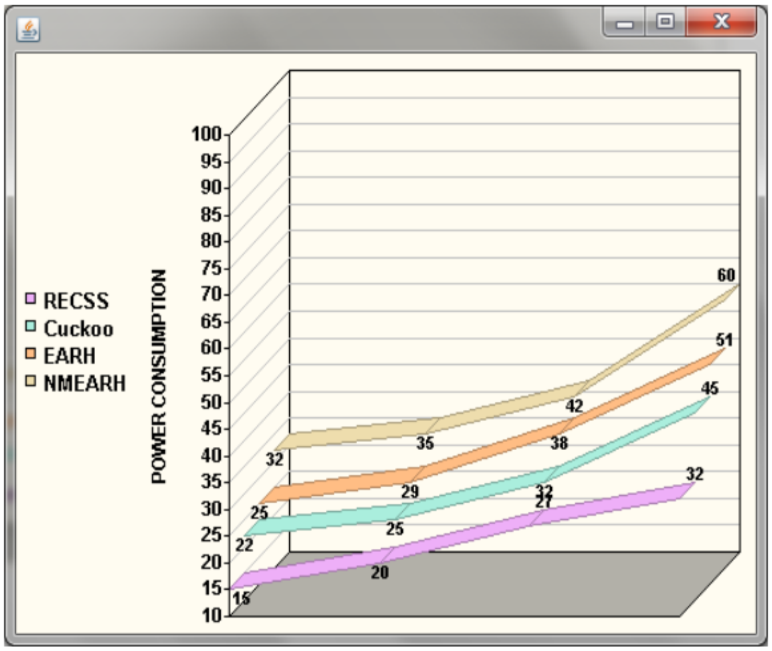

Fig. 3: Task count vs. power consumption

considerably when the task count increases. The CS algorithm and NMEARH does not employ the VM migration and does not employ rolling-horizon optimization which leads to high percentage of resource allocation when compared to RECSS.

Comparison of RECSS with NMEARH, EARH and CS algorithm for Power Consumption: Figure 3 show the performance of the three algorithms on power consumption with RECSS. It was observed that the proposed algorithm minimizes power when compared to

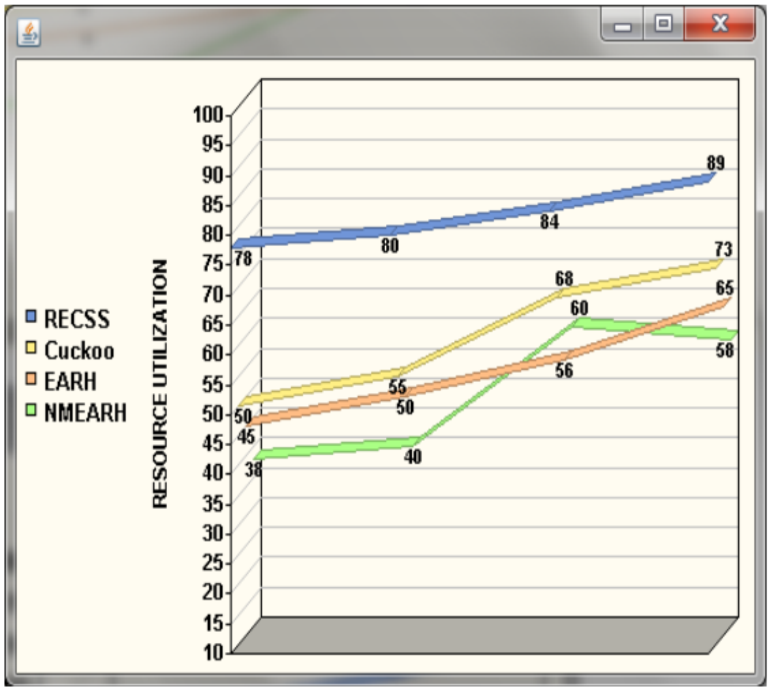

Fig. 4: Task count vs. resource utilization

NMEARH, EARH and CS. The power consumption of CS is higher than RECSS because there are unlimited resources in clouds, thus when task count increases, new hosts will be in progress by the CS to finish more tasks. However, not all the tasks can be completed effectively although, there are sufficient resources. One can point this to the fact that starting a new host or creating a new VM needs extra time cost which makes some concurrent tasks with firm deadlines cannot be finished within their timing limitations. Thus, more resources are allocated to the new host which leads to high cost and high power consumption in NMEARH, EARH and CS. Besides, RECSS have low power consumption because it employs rolling-horizon optimization policy that is able to make tasks with firm deadlines terminate earlier. Therefore, the schedulability is significantly improved in RECSS.

Comparison of recss with nmearh, earh and cs algorithm for resource utilization: Figure 4 it was observed that the proposed RECSS has much higher resource utilizations than the other three algorithms. This is because the utilization of rolling-horizon policy in RECSS makes the system acknowledge more tasks with firm deadlines which sometimes needs new computing resources and thus minimizes the resource utilization a bit. Conversely, the VM migration policy can make the system entirely utilize the host computing capacity. Because of non-migration of VM in NMEARH, the resource utilization is very low.

Comparison of RECSS with NMEARH, EARH and CS algorithm for execution time: Figure 5 shows the comparison of execution time of the proposed RECSS algorithm with the other three algorithms. It was clear from Fig. 5 that the execution time is low in RECSS than NMEARH, EARH and CS. This is because highly secured VM migration is possible in RECSS. The execution time is high in NMEARH than EARH and CS due to non-migration of VM which takes a lot of time to execute the tasks.

Overall performance evaluation: Figure 6 shows the overall performance analysis of the proposed system with CS, EARH and NMEARH. The RECSS improves the overall performance with increase in task count compared with CS and EARH. NMEARH degrades the overall performance because of non-migration. The overall performance of NMEARH increases rapidly with low task count but decreases with increase in task count.

Table 1 illustrates that RECSS show significant improvement when compared with NMEARH, EARH and CS algorithms. It can be observed that the NMEARH preserves more energy, i.e., NMEARH has the most power consumption when compared to EARH, CS and RECSS. It indicates that utilizing the VM migration policy is very efficient when scheduling concurrent tasks. On one point of view when the task count increases, current VMs can be combined to make some room for creating new VMs

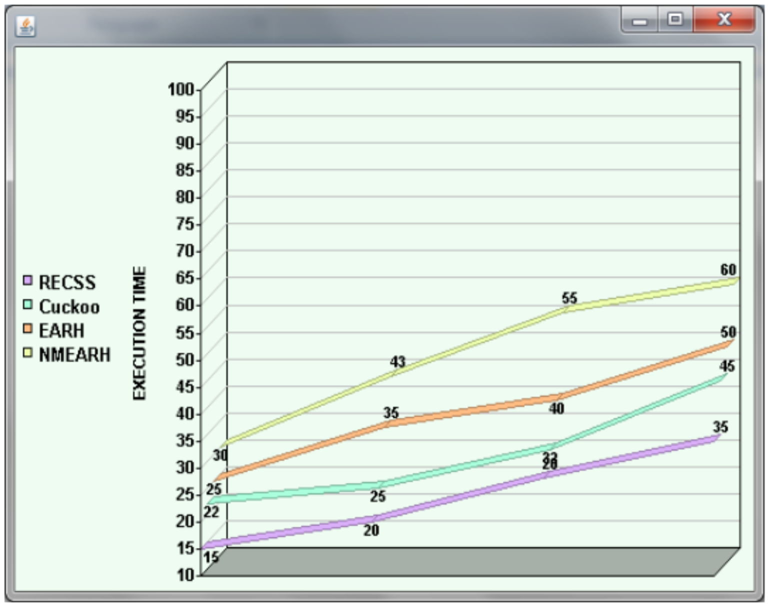

Fig. 5: Task count vs. execution time

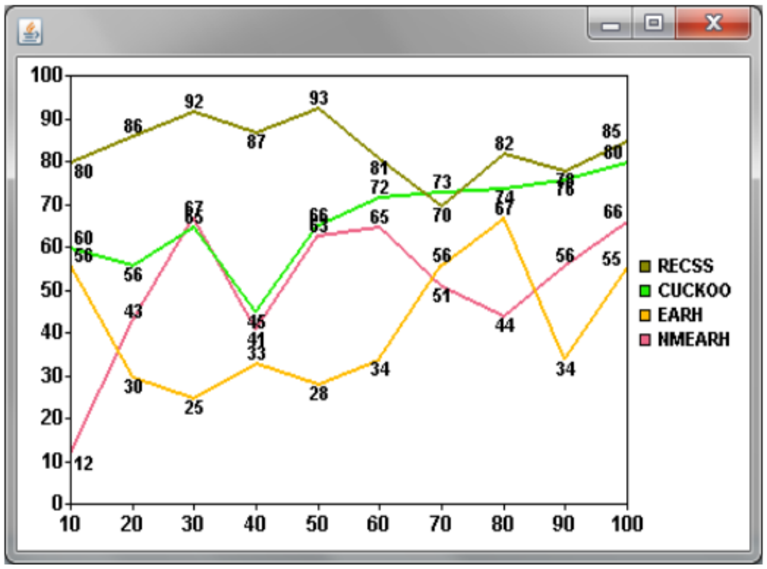

Fig. 6: Overall performance analysis

Table 1: Performance metric evaluation

Performance metrics	RECSS	Cuckoo search	EARH	NMEARH
Resource allocation (%)	20	30	33	39
Power consumption(*10^6) W	15	22	26	32
Resource utilization (%)	78	50	45	38
Execution time (sec)	15	25	29	35

which avoids the power consumption caused by adding new active hosts. On the other hand, the VMs in light-load host can be migrated to other hosts and then the inactive hosts can be shut down which further reduces the energy consumption.

CONCLUSION

This research presents a comparative analysis of various scheduling algorithms such as RECSS, EARH and NMEARH, taking into consideration the energy awareness for optimal performance of cloud data centers and achieves excellent provisioning of resources. The algorithms focus on various parameters such as resource allocation, resource utilization, power consumption and execution time. The evaluation shows the proposed RECSS algorithm outperforms significantly the other two algorithms.

RECOMMENDATIONS

The future research aims to compare the proposed algorithm with Cuckoo search algorithm and to obtain the experimental results based on the metrics considered in this research.

REFERENCES

Barroso, L.A., J. Clidaras and U. Holzle, 2013. The datacenter as a computer: An introduction to the design of warehouse-scale machines. Synth. Lectures Comput. Archit., 8: 1-154.

Calheiros, R.N., R. Ranjan, A. Beloglazov, C.A.F. de Rose and R. Buyya, 2011. CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Pract. Experience, 41: 23-50.

Chiesi, M., L. Vanzolini, C. Mucci, E.F. Scarselli and R. Guerrieri, 2015. Power-aware job scheduling on heterogeneous multicore architectures. IEEE. Trans. Parallel Distrib. Syst., 26: 868-877.

Fan, X., W.D. Weber and L.A. Barroso, 2007. Power provisioning for a warehouse-sized computer. Proceedings of the ACM SIGARCH Computer Architecture News Vol. 35, June 09-13, 2007, ACM, San Diego, California, USA., ISBN:978-1-59593-706-3, pp: 13-23.

Goiri, I., F. Julia, R. Nou, J.L. Berral and J. Guitart *et al*., 2010. Energy-aware scheduling in virtualized datacenters. Proceedings of the 2010 IEEE International Conference on Cluster Computing (CLUSTER), September 20-24, 2010, IEEE, Heraklion, Crete, Greece, ISBN:978-1-4244-8373-0, pp: 58-67.

Hsu, C.H. and W.C. Feng, 2005. A power-aware run-time system for high-performance computing. Proceedings of the 2005 ACM-IEEE Conference on Supercomputing, November 12-18, 2005, IEEE, Washington, DC, USA., ISBN:1-59593-061-2, pp: 1-1.

Jeba, J.R. and D.S. Victor, 2011a. Comparison of frequent item set mining algorithms. Intl. J. Comput. Sci. Inf. Technol., 2: 2838-2841.

Jeba, J.R. and S.P. Victor, 2011b. A novel approach for finding frequent item sets with hybrid strategies. Intl. J. Comput. Appl., 17: 30-33.

Petrucci, V., O. Loques and D. Mosse, 2009. A dynamic configuration model for power-efficient virtualized server clusters. Proceedings of the 11th Brazillian Workshop on Real-Time and Embedded Systems Vol. 2, May 25, 2009, National Science Foundation, Virginia, USA., pp: 35-44.

Wang, X., M. Chen, C. Lefurgy and T.W. Keller, 2012. Ship: A scalable hierarchical power control architecture for large-scale data centers. IEEE. Trans. Parallel Distrib. Syst., 23: 168-176.

Wu, L., S.K. Garg and R. Buyya, 2012. SLA-based admission control for a software-as-a-service provider in cloud computing environments. J. Comput. Syst. Sci., 78: 1280-1299.