International Journal of Computational Intelligence Research
ISSN 0973-1873 Volume 13, Number 1 (2017), pp. 45-53

http://www.ripublication.com

Power-Aware Scheduling For Urgent Tasks in Cloud Environment

Ankit Patel
PG student[1]

Dr. Hiren Patel
Professor[2]

Prof. Nimisha Patel
Associate Professor[1]
Research Scholar[3]

[1] *S.P College of Engineering Visnagar-Gujarat, India*
[2] *LDRP Institute of Technology & Research, Gandhinagar-Gujarat, India*
[3] *Rai University, Ahmedabad-Gujarat, India*

Abstract

Cloud computing has been an emerging technology for the last few years in which computing services and resources are made available to users on demands through the Internet on a rental basis. Increased usage of Cloud has resulted into significant amplification in a number of Cloud data centers and in turn huge amount of energy consumption. Therefore several researchers have drawn their attentions in addressing the issue of energy consumption of Cloud data centers. In this work, we aim to address the issue of reducing the power required to execute urgent or high-priority tasks. Based on various factors such as level of urgency, task deadline, task runtime, VM reusability and suspension of a non-urgent task, we define the priority of task and accordingly assign the task to suitable virtual machines (VMs) on a compatible host. We propose a modified power-aware scheduling for urgent tasks that combines Dynamic Voltage Frequency Scaling (DVFS) and VM Reusability. The proposed method aims to reduce power-consumption while maintaining

availability for priority-tasks without compromising commitments to the user. In future, we plan to simulate the proposal on CloudSim and compare it with existing techniques for checking its feasibility and measure the enhancements.

Keywords: Cloud computing, Urgent Task Scheduling, Energy Consumption, VM Reuse, DVFS.

1. INTRODUCTION

As information and communication technology (ICT) grows gradually, need for the computational resources has also increased than before. The autonomy of a huge amount of the IT assets may not be a good alternative to many industries, so now IT industries lease rather than own Cloud applications and services. Combining the features of a grid, cluster and utility computing mingled with virtualization, Cloud computing offers an innovative model of "computing as a utility". Currently providing dynamic services like processing elements, memory, storage, bandwidth etc. be a provision on demand over the Internet on a pay-per-use basis of elasticity and flexibility [1]. Cloud computing has been coined as an umbrella term to describe the category of sophisticated on-demand computing services initially offered by IT giants including Microsoft, Google, Apple, IBM etc. It denotes a model on which computing infrastructure is viewed as a Cloud from which business and individuals access applications from anywhere in the world on demand. [2] Many researchers have paid particular attention to research in security and energy consumption in the field of Cloud computing. The issue of energy consumption can be addressed to many directions but, we aim to address the issue using task scheduling mechanism for an application having varying priority.

As defined by [1], Cloud computing describes three types of service models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). In recent years, due to massive use of Cloud, a large amount of data centers consume huge energy. Many researchers have now paid attention to green computing for reducing energy consumption. Teena Mathew et al.[3] describe there are many scheduling techniques available in recently such as FCFS (First Come First Served), Minimum Execution Time Algorithm (META), Task Scheduling and Server Provisioning (TSSP), Priority Task Scheduling Algorithm, User Priority guided Min-Min, WLC-based scheduling, Enhanced Max -Min Algorithm, etc. Efficient usage of scheduling the algorithm may result in proper workload distribution and execution which in turn assist in saving power consumption. Scheduling can also be done for the specialized task such as an urgent job.

The rest of the article is organized as follows. Section II discusses related work pertaining to the domain and various existing scheduling techniques that have been carried out in the area over recent years in the Cloud computing. Section III discusses our proposal regarding the planning of Power-Aware urgent tasks and the reduction of energy consumption. We describe modified Power-Aware Energy Efficient

Scheduling in this section. We conclude our work in Section IV and list the references used in Section V.

2. RELATED WORK

Task scheduling is multi-objective constrained optimization problems. Objectives of task scheduling are to reduce the energy consumption, higher urgency tasks executed first, finish tasks before user defined deadlines, scheduled tasks without compromise QoS of tasks along with better resource utilization and improve a performance of the system.

Buyya et al. [4] define urgent task scheduling combined with energy saving technique DVFS (Dynamic Voltage Frequency Scaling). Authors use Cloud-Aware Energy Efficient Scheduling Algorithm for finding best VM for given task. New tasks first arrived at RMS (Resource Management System). All tasks are arranged in non-decreasing order for task deadline. Higher urgency tasks allocate first to VM. RMS decide which task is to be executed on which VM. Criteria to choose suitable VM are (and in order of) (a) the task is first scheduled for those VMs which is already in use and does not require frequency to be increased (b) the task is scheduled for those VMs which is already in use, but CPU frequency has to be increased. (c) the task is scheduled to host with at least one VM in use. (d) the task is scheduled to idle VM as well as idle host. Authors take two urgency jobs (i) higher urgency and (ii) low urgency jobs. They also generate results and compare them with baseline algorithms. At the end, authors claim that the proposed algorithm reduces the energy consumption of urgent task scheduling. As part of future work, authors put forward to progress algorithms to carry other types of application likes 1) works flows 2) Map Reduce. Abdulrahman et al. [5] present new Energy-Aware Task scheduling frameworks that used two well-accepted techniques DVFS and VM Reusability in Cloud data center. The algorithm consists three phases: (i) Scheduling policy setting- sorting all PE (Physical Element) according to scheduling policy such as FFD (First Fit Decreasing) and WRR (Weighted Round Robin). (ii) DVFS utilization- DVFS is modern energy saving technology that maintains CPU operating at a minimum voltage level. (iii) VM Reuse - VM Reuse technique improved resource utilization. VM Reuse a model first verifies any new incoming task to be scheduled to find best fit reusable VM, then check whether it has appropriate extension gaps (backward or forward) for given task. Authors take some VM and describe different possible cases of possible VM reusability. Authors proposed Energy-Aware Task Scheduling- First Fit Decreasing (EATS-FFD) algorithm and using ClouSim, their simulation results are compared with Enhanced First Fit Decreasing (EFFD), Enhanced Weighted Round Robin (EWRR) and Energy-Aware Task Scheduling-Weighted Round Robin (EATS-WRR). Authors claim this resource utilization model is more accurate and suitable for Cloud data centers than traditional models. SaeMi Shin et al. [6] proposed a scheduling algorithm to develop both the deadline and resource utilization. Authors introduce modified conservatives backfilling function algorithms by exploiting the earliest deadline first (EDF) and largest weight first (LWF). In proposed algorithms,

all tasks arrive at the datacenter (DC) and system considers tasks to be first allocated to the resource manager (RM) and put tasks into waiting for queue. It then sorts all tasks in a non-descending order to allocate primary and emergency tasks compared to low urgency tasks. RM also monitors VM status (idle, busy) in the data center. Job manager (JM) gets information about tasks and updates when a new task arrives. Using the information on JM and RM, the broker checks if tasks can be allocated or not. Propose the algorithm then chooses the greatest backfill work possible to ensure the deadline. At the end of simulations, authors claim that the algorithm is very efficient and task migration is carried out efficiently to get better performance. Scheduling of real-time tasks is identified as a major concern to present Quality of Service (QoS) by Menglan et al. [7]. Authors propose dynamic real-time scheduling referred as scheduling the algorithm with migration (SAM) for managing Bag of Tasks (BoT) application. SAM uses EDF (Earliest Deadline First) to choose best-fit tasks in the BoT queues. But it not clarified which processor will be assigned tasks to solve the problem. SAM algorithm is classified into 3 parts: (i) SAM algorithm allocates tasks with Max-Min policy (ii) Minimum numbers of processors used for tasks so that energy put aside for unplanned tasks. (iii) SAM also exploits task migration for load balancing if tasks may finish late and new tasks arrive. Authors compare the algorithms with conservative backfill algorithm and claim with simulation results that the proposed algorithm is better in deadline miss-rate and better way to use resource utilization. George et al. [8] study BoT load balancing with an energy point of view and propose two load balancing mechanisms that are estimated at the simulation in Cloud environments. LCP (Lowest Completion Time) that moves tasks based on minimum completion percentages of an idle processor and HCP (Highest Completion Time) that moves tasks based on maximum completion percentage to an idle processor. BoTs and high priority real time tasks are both assigned to the cluster and more accurate workload put forward to the system. Authors claim to save the significant amount of energy and that LCP achieves higher energy saving than HCP.

3. OUR PROPOSAL

Our proposed the system model is illustrated in Figure 1, which is composed of virtualized IaaS data center and supports the PaaS layer that supports urgent task scheduling. The data center is composed of a number of hosts and each host comprises of a number of VMs. Every host has VM Manager to manage VMs running on the host. RMS is primarily the role of assigning tasks to the appropriate virtual machine from different users. RMS is responsible for supply and manages resources along with the handling of incoming requests. For the incoming tasks to be placed on VM, for each VM a separate queue is maintained to store the tasks until they are not placed on the VM. At any point of time, only one task is executed and the remaining tasks are kept in the queue. In proposed model, all tasks are meant to be executed before user defined deadlines. Deadlines are not supposed to be missed. If deadlines are overlooked, the perception of energy consumption is considered exhausted. Therefore RMS verifies that if deadlines are not met, the request is discarded. When a

new task arrives in the system, various factors are to be decided such as (a) which VM can serve the task, (b) where the task is to be placed in a given queue and (c) what frequency level is to be applied to the VM.

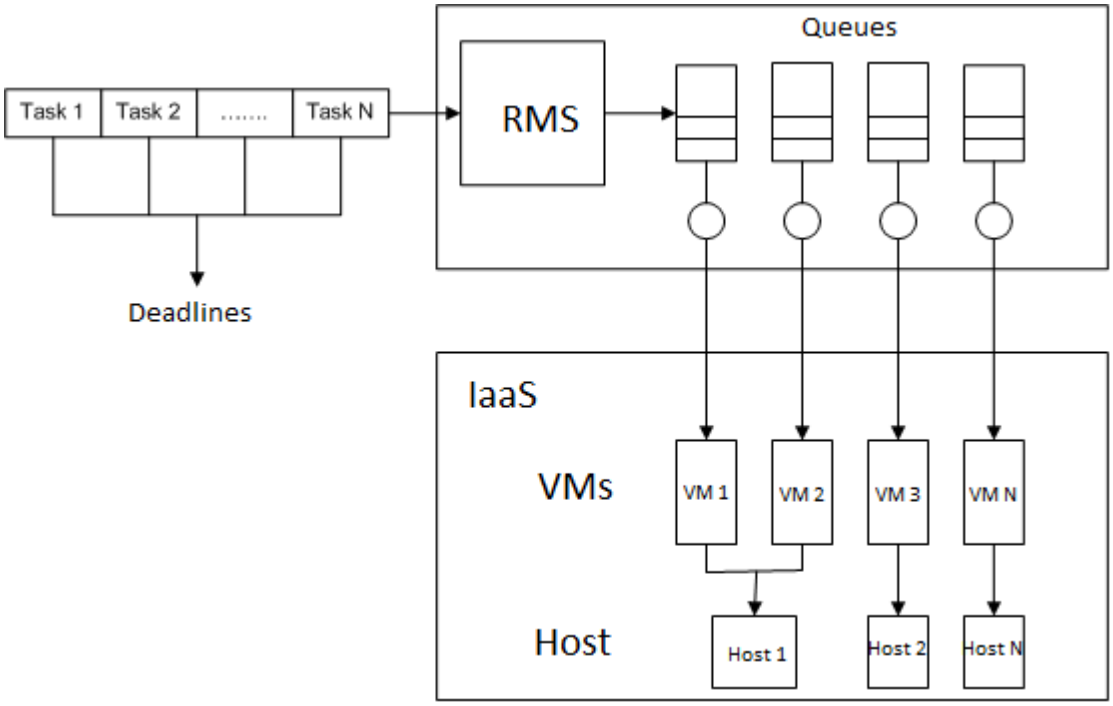

Figure 1. System Model

In our proposed model, we used two already known techniques 1) DVFS 2) VM Reuse. Dynamic Frequency and Scaling (DVFS) is a modern CPU processor to maintain CPU in use of less amount of voltage level for the power consumption achieve before user deadline. DVFS is a technology that reduces energy consumption. For each task submitted, it calculates the value of an appropriate processor as well as VM that executes the task. We also use the concept of VM Reuse where each task is submitted to the local manager (LM) of the host being processed that ensures that the VM list is kept to finding the most appropriate VM Reuse machine for the given tasks. If an appropriate match is found, the task is assigned to this VM, otherwise to a newly launched VM. In our main objective of scheduling the urgent tasks considering energy consumption, we also added the concept of suspension of tasks. For the process of performing non-urgent tasks if a new incoming urgent task has arrived, we have priority for urgent tasks. The non-urgent task is suspended and placed last in the queues and it executes incoming urgent tasks. When the urgent task is completed, a non-urgent task is resumed. Based on all these facts, we propose Modified Power-Aware Energy-Efficient Scheduling algorithm which is presented below after the flowchart.

Flowchart :

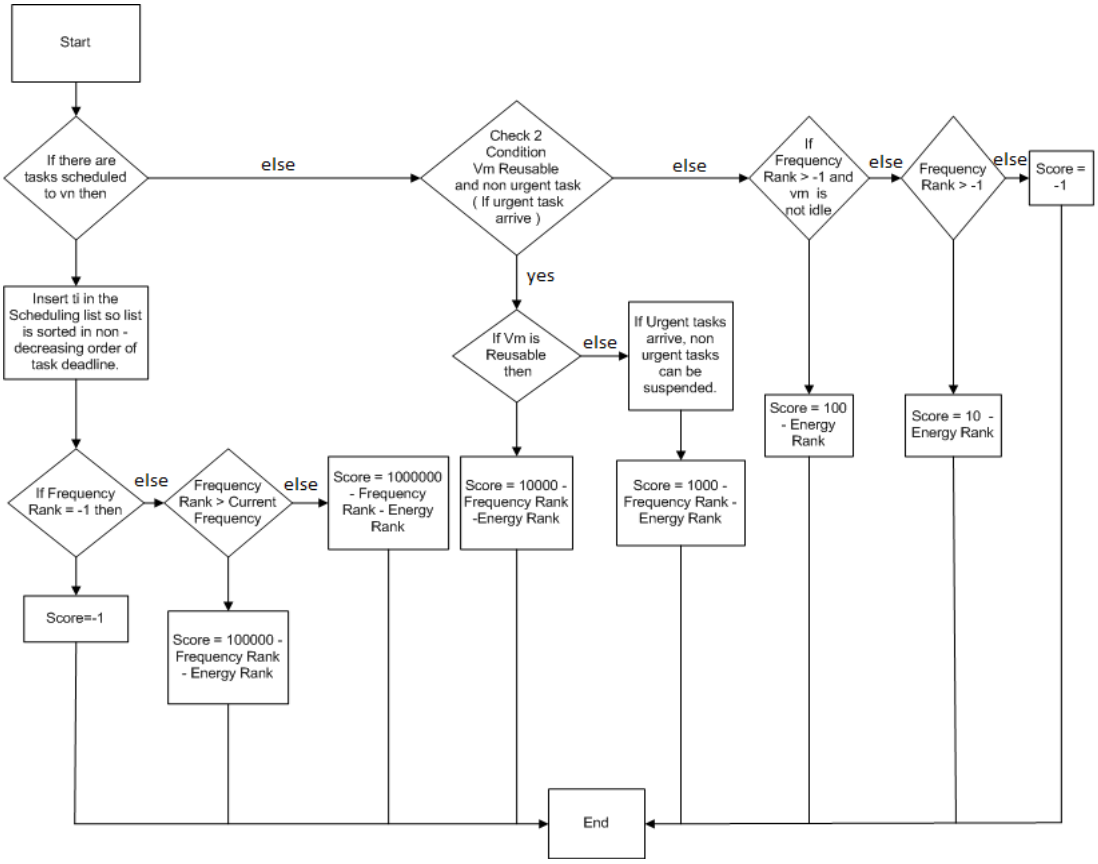

Figure 2. Flowchart of our Proposed Method

.

Fig. 2 describes our proposed method which illustrates different possibilities to choose best fit VM for the given tasks with deadlines.

Algorithm: Modified Power-Aware Energy-Efficient Scheduling :

1:**for each** task *ti* in *J* **do**
2: *Vm* ← null
3: *Position* ← null
4: *Frequency* ← null
5: Task Urgency=task urgency T
6: *max Score* ← 0
7: **for each** *vm* in *VM* **do**
8: *score* ←0; *position* ←0; *freq Rank* ←0;
9: *energy Rank* ← energy ranking of the host where the VM runs;
10: **if** there are tasks scheduled to *vm* **then**

```
11:             Insert ti in the scheduling list so that the list is sorted in non- decreasing order of task
                    deadline;
12:             position ← position of ti in the scheduling list;(position of  Ti after sorting )
13:             freq Rank ← index of the smallest frequency level able to meet the deadline of all
                            tasks in the scheduling list, or -1 if no frequency meets all the deadlines.
14:                             if  frequency Rank = −1 then
15:                             score = −1;
16:                     else
17:                             if freq Rank > current freq of  Vn then
18:                                     score ← 100000 − freq Rank − energy Rank;
19:                             else
20:                                     score ← 1000000 − freq Rank −energy Rank;
21:                             end if
22:                     end if
23:                     else
24:                     if  Vm reusable then
25:                             score ← 10000 − freq Rank − energy Rank;
26:                     else
27:                             for each task T scheduled in list do
28:                                     if Urgency(T) < Urgency(Ti) then
29:                                             Suspend T
30:                                             add T to Task List in Last
31:                                             exit for
32:                                             end if
33:                             end for
34:                     score ← 1000 − freq Rank −energy Rank;
35:             end if
36:             else
37:   freq Rank ← index of the smallest frequency level able to meet the deadline of   ti, or -1 if no
                        frequency meets the deadline;
38:             if freq Rank > −1 and the host where Vn runs contains at least 1 VM not idle then
39:            score = 100 − energy Rank;
40:       else if frequency> −1 then
41:            score = 10 − energy Rank;
42:       else
43:           score = −1;
44:    end if
45:             Insert ti in the empty scheduling list for vn;
46:             position ← 0;
47:    end if
48:             if score > max Score then
49:                     max Score ← score;
50:                     chosen Position ← position;
51:             chosen Frequency← frequency Rank;
52:             chosen Vm ← vn;
53:       end if
54:    end for
55:       if chosen V m = null then
56:             Remove all scheduled tasks from J from the scheduling queues;
57:             Return failure.
58:       end if
59: end for
60: Return success.
```

The main objective of the proposed algorithm is the management of priority based or urgent tasks. Such tasks are executed at the best suitable VM within the deadlines defined by the users and by reducing energy consumption. RMS allocates the task given to the appropriate VM and it allocates the task to the 6-way decision possibility described below.

The modified Power-Aware Scheduling algorithm chooses the most appropriate VM for each task. It retains the idea of marking and selecting the energy rating of the system host. The possible decision to select the best appropriate VM for a given task are (and in order of): (1) Scheduled the tasks to those VMs that is already in use and do not require improvement in frequency (2) Scheduled the tasks of the VM that is already in use, but the processor frequency needs to be improved (3) Verify the possibility of VM Reuse from above two situations and if found in that case schedule the task to the VM (4) If a new incoming urgent task has arrived, suspend the non-urgent task and execute the urgent tasks to that virtual machines (VM). A suspended job is placed last in the queue. When an urgent task is completed than to resume the non-urgent suspended task. (5) Scheduled tasks to the virtual machine, but the host has at least one VM in service (6) Scheduled the tasks to the inactive VM as well as the inactive host. The only requirement is the value assigned to the score is greater than the maximum value that can be achieved by the next choice. If the frequency level is less than or equal to 10, the base score of the decision is 1000000, 100000, 10000, 1000, 100, 10. If there is a negative value of the score, then there is no frequency correspondence with deadlines. Lines of algorithms 18, 20, 25, 34, 39 and 41 never result in a negative score value or cause the value of the score next possible scheduling decision to be taken. Since if the previous score is negative then the frequency of VM does not meet deadlines.

Energy ranking is given by the efficiency of the host that is most efficient host gets higher ranks. In the algorithm the most efficient host is ranked 0, the second efficient host is given the rank 1 and so on. For the concept to be applied, each task is repeated (lines 7-54). It checks the first VM status of utilization. If VM in use (line 11), the task is inserted into non-empty execution queue of VM. The arrangement of the task is decided by non-decreasing order for the given deadlines. If VM is not suitable to meet the deadlines, VM is set to -1 (line 15). If VM can be used without an increase in frequency (line 20) otherwise with an increased frequency of VM (line 18), the base score 1000000, 100000 respectively. If VM can be reused for the 2 upper sections (line 25), the score is 10000. If the urgent task has arrived and it has to suspend non-urgent tasks (line 34) then the score is 1000. Assignment of the score 100 if the host is used but VM is inactive (line 39). If the host is also inactive with VM, the score value is 10 (line 41) otherwise score is -1 (line 43). No deadlines correspond to the frequency. The selection of the best VM tasks occurs between lines 48 to 53. Since maxScore is initially set to 0, the negative value does not lead to the selected VM update. This means that no VM should meet the deadline, choosenVm will be initially zero and will be a failure (line 55-57). In this case, the scheduled task removed from

the queues and the task is rejected. If all tasks are successfully scheduled, the process is successfully called (line 60) and the task is accepted for the execution.

4. CONCLUSION

In this work, we aim to address the issue of urgent task handling while maintaining energy efficiency and without compromising commitments to the user. We have proposed a modified power-aware scheduling for urgent tasks that combines Dynamic Voltage Frequency Scaling (DVFS) and VM Reusability to address various factors such as level of urgency, task deadline, task runtime, VM reusability and suspension of a non-urgent task. In our next phase of our future work planning, we plan to simulate our mechanism into CloudSim and compare our method with existing ones.

REFRENCES

[1] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," National Institute of Standard and Technology, Information Technology Laboratory 800-145, September 2011.

[2] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, Ivona Brandic "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility" Future Generation Computer Systems, 25:599-616, 2009.

[3] Teena Mathew, K. Chandra Sekaran, John Jose "Study and Analysis of Various Task Scheduling Algorithms in the Cloud Computing Environment". 978-1-4799-3080-7/14 © 2014 IEEE

[4] Rodrigo N. Calherious and Rajkumar Buyya "Energy-Efficient Scheduling of Urgent Bag-Of-Tasks Application in Cloud through DVFS." 978-1-4799-6/14 © 2014 IEEE.

[5] Abdulrahman Alahmadi, Dunren Che, Mustafa Khaleel, Michelle M. Zhu, Parsia Ghodous. "An Innovative Energy-Aware Cloud Task Scheduling Framework." 2159-6190 © 2015 IEEE.

[6] SaeMi Shin, Yena Kim and SuKyoung Lee "Deadline-Guaranteed Scheduling Algorithm with Improved Resource Utilization for Cloud Computing." 978-1-4799-6390-4/15 © 2015 IEEE.

[7] Menglan Hu and Jun Luo." Dynamic Real-Time Scheduling with Task Migration for Handling Bag-of-Tasks Applications on Clusters." 978-1-4673-4523-1/12 © 2012 IEEE.

[8] George Terzopoulo, Helen D.Karatza "Bag-of-Tasks load balancing on power-aware Clusters." 978-1-4673-8776-7/16 © 2016 IEEE.

[9] Calheiros R N, Buyya R, Beloglazov A, Rose CAFD, Buyya R. CloudSim: a toolkit for modeling and simulation of Cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience, Wiley Press, New York, USA, 2010.