Task Scheduling Techniques for Minimizing Energy Consumption and Response Time in Cloud Computing

M Dhanalakshmi
Dept of CSE
East Point College of Engineering & Technology
Bangalore, India

Anirban Basu
Dept. of CSE-R&D Centre
East Point College of Engineering & Technology
Bangalore, India

***Abstract*— Cloud computing which provides services "on demand" is gaining increasing popularity. However, one of the most challenging problems in cloud computing is to minimize the power consumption in the data centers. The subject of Green Cloud Computing has emerged with the objective of reducing energy consumption. While reducing the energy consumption is of importance to Cloud Service Providers, performing the computation in minimum possible time (makespan) is of interest to users. In this paper we propose a technique to achieve the twin objective of minimizing the energy consumption as well as reducing the makespan of tasks. A method has been proposed and its effectiveness verified by simulating on CloudSim[21]. Results presented in this paper show the advantages of the proposed technique.**

Keywords*— *Task Completion time, Task Execution time, Task scheduling algorithm, Min-min, Max-min, RASA (Resource Aware Scheduling Algorithm), Improved Max-min Algorithm, VM (Virtual Machine) Placement.

I. INTRODUCTION

Cloud Computing[1] makes computer infrastructure and services available "on-need" basis. It is believed that cloud computing will be dominant computing platform in the days to come. Virtualization software and Virtual Machine (VM) placement are important aspects in a Cloud Computing environment. A primary requirement to make a cloud computing environment attractive to an user is by reducing the response time by minimizing the makespan (i.e. the response time of jobs) [2]. However the huge energy consumption in data centers in a Cloud Computing platform is a cause for concern as it determines the number of jobs a Cloud Service Provider can handle at a time. The subject has received considerable attention. Termed as Green Cloud Computing, it seeks to minimize energy consumption in a Cloud Computing environment [3][4]. Many task scheduling algorithms are being used in cloud computing environment to optimally allocate resources to tasks to satisfy QoS requirement specified in Service Level Agreements (SLA). The amount of energy consumed is determined by the size of the data center in terms of the number of servers. Therefore for sustaining the growth of cloud computing data center[5] resources need to be managed in an efficient manner[6] and cloud resources need to be allocated efficiently to reduce energy usage. However the most of the proposed scheduling algorithms do not take into account both the aspects of energy consumption and makespan. In this paper the objective is to focus on developing a scheduling technique which combines Virtual Machine (VM) placement algorithm and modified version of Max-min algorithm to minimize both energy consumption and makespan.

II. RELATED WORK

One of the power management and performance maximization techniques applied at the data center level, was developed by Jonathan et al. [7]. In this work, the authors have proposed a technique for minimization of power consumption and improving performance in a heterogeneous cluster of computing nodes. A Cloud is seen as a shared and distributed environment, where concurrent users run VMs on it. However when multiple Virtual Machines share the same physical resource, the performance of each VM and its embedded application depends on the resource usage of other VMs running on the same physical host. So management of VMs becomes critical. The technique is applied by considering objective of the users: some users run critical programs and others want cheap computing. The scheduling algorithm performs VM placement and reconfiguration based on the user expression of quality, trust and protection. It achieves the goal of energy reduction and performance maximization.

Xin Li ^{a} et al. [8] have considered the problem of energy-efficient virtual machine placement algorithm with balanced and improved resource utilization in a data center. The main challenge is to improve the resource utilization to minimize power consumption. The key issue in server virtualization is Virtual Machine Placement (VMP) which selects some available physical machine (PM) to deploy each newly created VM in runtime. When the VMs are fully loaded in PMs, all the utilized resources referred to as resource fragments are wasted. In order to improve the resource utilization such that the number of running PMs is minimized, we have to lower the number of resource fragments and decrease their sizes. To achieve this authors propose a multidimensional space partitioned model to characterize the usage of each PM. This algorithm reduces the number of resource fragments and decrease their sizes and minimize the energy consumption.

Saeed Parsa et al. in [2] have proposed a task scheduling algorithm based on a comprehensive analysis of the two well known task scheduling algorithms Min-min, Max-min. He defines RASA (Resource Aware Scheduling Algorithm) using the advantages of Min-min and Max-min algorithms. RASA estimates the completion time of the tasks on each of the available grid resources and then applies the Max-min and Min-min algorithm.

O.M. Elzeki et al. in the paper [9], have discussed an algorithm based on RASA algorithm. An improved version of Max-min algorithm is proposed to outperform scheduling. Improved Max- min algorithm is based on the expected execution time instead of completion time as the basis of selection. This algorithm achieves lower makespan than RASA and original Max-min algorithm.

Upendra Bhoi et al.[10] in their paper, have proposed the enhanced Max-min algorithm based on the expected execution time. The only difference is that Improved Max-min algorithm assigns task with maximum execution time (i.e., the largest task) to slowest resource, while enhanced Max-min algorithm assigns task with average execution time to resource produces minimum completion time.

There are many other VM placement algorithms[11-15] that have been proposed. However, none of the algorithms focus on the problem of minimizing energy consumption and response time. In this paper, an algorithm is proposed which is more efficient compared to the above discussed energy efficient algorithms. Most of the earlier algorithms considered only the CPU power, but in the proposed algorithm for reducing energy consumption, the VM placement algorithm considers the power of CPU, RAM and Hard Disk. Based on this the number of servers which will be switched off will increase results in reducing energy consumption. The modified version Max-min algorithm is used to minimize the response time. The technique proposed is discussed in section III.

III. PROPOSED TECHNIQUE

We now discuss a technique for Task Scheduling which reduces both the energy consumption and makespan of the tasks. The proposed method is a combination of VM placement algorithm to reduce energy consumption and modified Max-min algorithm (which aims to reduce makespan).

A. Reducing Energy Consumption

Reducing energy consumption is achieved by VM Placement algorithm which considers the energy consumed in CPU, RAM, and Hard Disk.

1) Energy Equation: Earlier algorithms considered only the CPU power for VM placement. In the proposed algorithm allocation of VMs is based on power of CPU, RAM and Hard Disk. The power model defined in [16,17] gives the energy consumed in CPU, RAM, and Hard Disk as specified below.

$$P_{CPU} = [\alpha, \beta \ldots \gamma] + \textbf{Vector of Performance Counters} + \tau_{constantCPU} \quad (1)$$

$$P_{RAM} = [\Delta, \Gamma, \ldots \theta] + \text{Vector of Performance Counters} + \tau_{constantRA.} \quad (2)$$

$$P_{Disk} = [\varphi, \dashv] + \textbf{Vector of Performance Counters} + \tau_{constantDisk} \quad (3)$$

we define The Total System Power as a function of the utilization of CPU, RAM and Hard Disk. If Psysmax is the maximum power consumed when the server is fully utilized; k is the fraction of power consumed by the idle server (e.g. 70%); u is the CPU utilization, r is RAM utilization and d is Hard Disk utilization then the total system power is as given below:

$$P_{sys} = P_{CPU} + P_{RAM} + P_{Disk} \quad (4)$$

$$Psys(u+r+d)=k.Psysmax+(1-k).Psysmax.u.r.d \quad (5)$$

Psysmax is normally 250 W, which is the normal value for modern servers. The utilization of the CPU RAM and Hard disk may change over time due to workload variability. The CPU, RAM and Hard Disk utilization is a function of time and is represented as u(t), r(t), d(t). Therefore, the total energy consumption by a physical node (E) can be defined as integral of the power consumption function over a period of time from t_0 to t_1 as shown in (6).

$$E = \int_{t0}^{t1} P_{sys}(u(t) + r(t) + d(t))dt \quad (6)$$

B. Proposed VM Placement Algorithm

In the proposed VM placement algorithm, VM (Virtual Machine) placement is done in two phases: the first phase deals with the admission of new requests for VM provisioning and switching off the nodes or hosts which do not meet the VM resource requirements (like CPU, memory, Hard Disk etc). The remaining physical nodes are ranked based on their resource availability. Higher rank is assigned to the node which has greater resource availability. In the second phase all the VMs are sorted in increasing order based on the power requirement of CPU, RAM, and Hard Disk. The amount of power consumed by the resources is related to the capacity used. Therefore, this ensures that the VM from the sorted list which provides least increase in power consumption is allocated to the node with higher rank. If the

CPU, RAM and Hard Disk requirement exceed the capacity of the node, then next VM from the sorted list is allocated to the next node in the rank list. The pseudo code for the algorithm is presented below.

Algorithm 1:

```
1. Input: nodeList,vmList,rankNodeList
2.Output : allocations of VMs
3.   foreach vm in the vmlist do
4.     foreach node in the nodeList do
5.      If node do not have enough resource for vm  then
6.         switchingoffNode()
7.rankNodeList.rankRemainingNode()
8.vmList.SortinIncreasingOrderfpower()
9. foreach  vm in vmlist do
10.        powerMin ← MAX
11.        allocatedHost ← NULL
12.   foreach node in rankNodeList do
13    power ← requirePower(node,vm)
14.    If  power < powerMin then
15.        allocatedNode ← Node
16.        powerMin ← power
17. If  allocatedNode ≠ NULL then
18.      allocate vm to allocatedNode
19. return allocations
```

C. *Modified Max-min Algorithm*

The normal Max-min algorithm has been modified to reduce the response time [18][19][20]. The paper proposes a method which is a modification of these. The Max-min algorithm allocates task Ti on the resource Rj where large tasks have higher priority over smaller tasks. For example, if we have one larger task, the Max-min algorithm could execute many shorter tasks on the fastest resources concurrently while executing the large one on the slowest resource. The total makespan, in this case is determined by the execution time of the larger task. Sometimes largest task is too large compared to other tasks in Meta-tasks, in that kind of case overall makespan is increased because too large task is executed by slowest resource, while other tasks are executed by faster resource, the load is imbalance across resources. Therefore, instead of selecting largest task if we select the average task or nearest greater than average execution time then overall makespan is reduced and also balances load across resources.

The algorithm proposed calculates the expected execution times of the submitted tasks on each resource, then compares the tasks in meta tasks such that difference between execution time of larger task and remaining other tasks is less than or equal to 1. Then the task with the overall maximum expected execution time is assigned to a resource that has the minimum overall completion time. If above said condition fails then the algorithm selects task with average execution time is assigned to a resource that has the minimum overall completion time. Finally, this scheduled task is removed from the list of meta-tasks and all calculated times are updated and Max-min algorithm on the remaining tasks is applied again. The proposed algorithm increases the opportunity of concurrent execution of tasks on resources and results in reducing makespan. The pseudo code for the modified Max-min algorithm is presented below.

Algorithm 2:

```
1. For all submitted tasks in meta-tasks; Ti.
   1.1    for all resources; Rj
        1.1.1Completion time; Cij =  Execution time; Eij + rj
2. if (( maximum execution of the larger tasks
      – execution time of remaining tasks in
     the meta-tasks)  <= 1 ) then
   2.1. Find task Tk costs maximum execution
        time.
   2.2Assign Tk to the resource Rj which gives
       minimum completion time
  else
   2.3 Find task Tk costs average or nearest
       greater than average execution time
   2.4 AssignTk to the resource Rj which gives
       minimum completion time
3. Remove task Tk from meta-tasks set.
4. Update rj  for selected Rj.
5. Update  Cij for all j
6. While meta-task is not Empty
    6.1 Find task Tk costs maximum
         completion time.
    6.2 Assign task Tk to the resource Rj
          which gives minimum execution time.
    6.3 Remove Tk from meta-tasks set
    6.4 Update rj for selected Rj
    6.5 Update Cij for all j
```

IV. EXPERIMENTAL RESULTS

The performance of the proposed method is measured by simulating using CloudSim[21].

A. *The VM placement algorithm to reduce energy consumption*

In Earlier techniques for VM allocation only CPU was considered but in the proposed method for reducing energy consumption we are considering CPU, RAM, and Hard Disk requirements. In our algorithm for allocating VM we consider all the resources CPU, RAM, Hard Disk. The nodes which do not meet the VM resources requirements (in terms of CPU, RAM, Hard Disk) are switched off and the energy consumption of these nodes is reduced as shown in Fig. 1. All VMs are sorted in increasing order of their CPU, RAM, Hard Disk power requirements. Each VM from the sorted list is scheduled on a node that provides the least increase in power consumption as shown in Fig. 2.

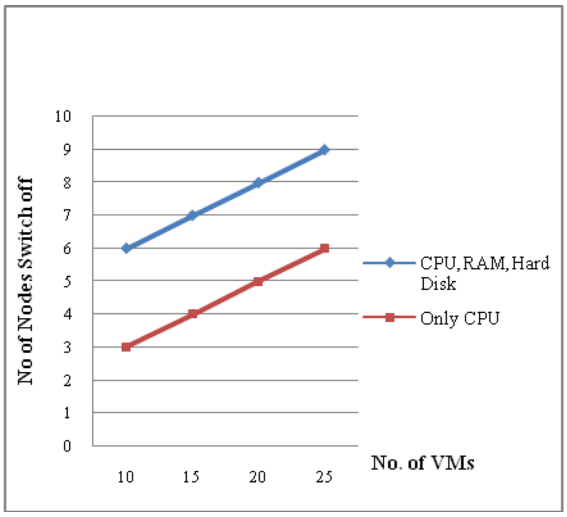

Fig. 1. CPU Vs CPU, RAM, Hard Disk for VM allocation

Table I. Makespan of jobs using proposed algorithm

Problem sample	Existing max-min Algorithm	Modified max-min Algorithm
P1	12	11
P2	10	9
P3	6	5
P4	5	4

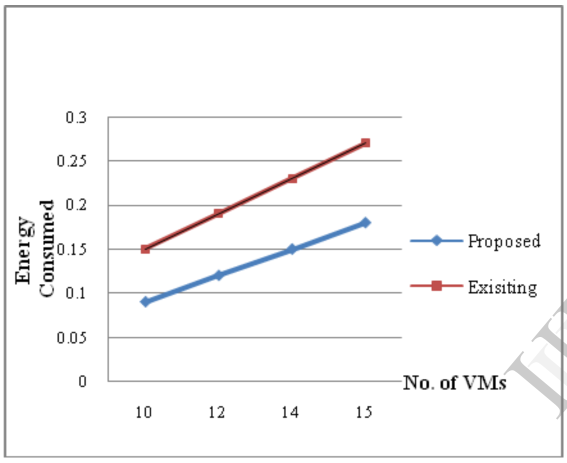

Fig. 2. Existing Vs Proposed Energy Aware Algorithm

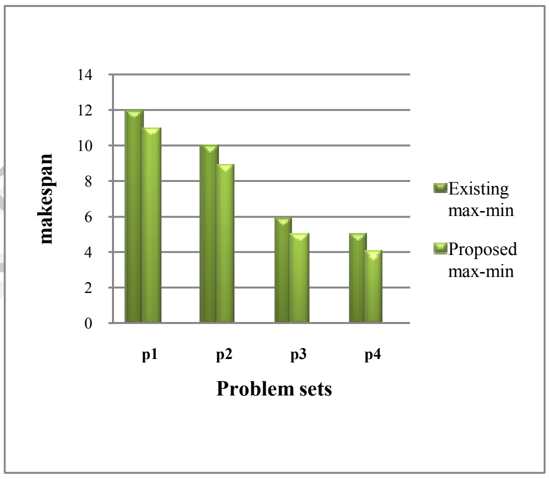

Fig. 3. Comparison of makespan

B. The modified version Max-min algorithm is used to minimize the response time.

In Modified Max-min algorithm, when we schedule the tasks, the task with maximum execution time (larger task) is assigned to the resources that produces minimum completion time (slower resources). In some cases larger task is too large compared to other tasks among the Meta tasks. In such case the overall makespan is increased. To overcome this situation the task with average execution time is assigned to the resources produces minimum completion time. This technique achieve shortest makespan compared to existing algorithm as given below in Table 1 and Fig. 3.

V. CONCLUSION

Existing scheduling algorithms do not consider both reducing energy consumption and makespan. Therefore there is a need to implement a new energy aware scheduling algorithm that can minimize the makespan as well as reduce the energy consumption to provide best possible solution. The paper presents a technique which achieves both objectives by using Virtual Machine (VM) placement algorithm to reduce energy consumption and modified version of Max-min algorithm to minimize the makespan. The experimental results simulated on CloudSim. It can be concluded that both the energy consumption and response time can be reduced by the proposed method.

REFERENCES

[1] Open Cloud Manifesto. [Online]. Available: http://www.opencloudmanifesto.org/.
[2] Saeedparsa and Reza Entezari-Maleki, "RASA: A New Grid Task Scheduling Algorithm", International Journal of Digital content Technology and its Applications,Vol.3, pp.91-99, 2009.
[3] The green grid consortium,2011
[4] EMC 2008 Annual overview Releasing the power of information, http:\\www.emc.comJ digital_universe.
[5] Ministry of Economy, Trade and Industry establishment of the Japan data center council, press Release.
[6] S.Albers, "Energy efficient Algorithms" Communications of the - ACM, Vol.53, no.5, pp.86-96, May 2010.
[7] Jonathan Rouzaud Cornabas, "A Trust ware Distributed and Collaborative scheduler for Virtual Machines in Cloud" May 11 2011
[8] Xin Li [a] , ZhuZhong Qian [a,*] , Sanglu Lu [a] , Jie Wu [b] ,"Energy efficient virtual machine placement algorithm with balanced and improved resource utilization in a data center", Feb 2013.
[9] O.M Elzeki, M.Z Reshad and M.A Elasoud, "Improved max min algorithm in cloud Computing", International Journal of Computer Application(0975 – 8887).
[10] Upendra Bhoi, Purvi N Ramanuj " Enhanced max min Task Scheduling Algorithm in Cloud Computing" International Journal of Application or Innovation in Engineering and Management, 6 April 2013.
[11] B.T. Benjamin Khoo, B. Veeravalli, T. Hung, and C.W. Simon See, "A multi-dimensional scheduling scheme in a Grid computing environment," Journal of Parallel and Distributed Computing, Vol. 67, pp. 659-673, 2007.
[12]http://opennebula.org/documentation:archives:rel2.0:Schg accessed on oct 30 2013.
[13] GWDG escience Group,"Virtual Machine allocation in current Cloud Computing Middleware",2012.
[14] Dailin Jiang, Peijie Huang, Piyuan Lin and Jiacheng Jiang, " Energy Efficient VM Placement Heuristic Algorithms Comparison for Cloud with Multidimensional Resources", ICICA 2012,LNCS 7473,pp 413- 420, 2012.
[15] Braun, T.D., H. Jay Siegel, N. Beck, L.L. Boloni, M Maheswaran, A.I.Reuther, J.P Robertson, M.D.Theys and B.Yao," A Comparison of Eleven Static Heuristics for Mapping a Class of Ondependent Tasks onto Hetergeneneous Distributed Computing Systems. Journal of Parallel and Distributed Computing", 61:810-837.
[16] DiPART (Disaggregated Power Analysis in Real Time) developed by Professor M. Srivastava, Y. Sun, and L. Wanner at University of California, Los Angeles.
[17] Y. Sun, L. Wanner, and M. Srivastava, Yuwen, Lucas and Mani. Low cost Estimaion of Sub-system power.
[18] Etminani, K and M. Naghibzadeh, "A Min-min Max-min Selective Algorithm for Grid Task Scheduling", The third IEEE/IFIP International Conference on Internet. Uzbekistan.
[19] He. X, X-He Sun, and Laszewski. G.V, "QoS Guided Min Min Heuristic for Grid Task Scheduling," Journal of Computer Science and Technology, Vol. 18, pp. 442-451, 2003.
[20] L. Mohammad Khanli, and M Analoui "Resource Scheduling in Desktop Grid by Grid-JQ999A," The 3rd International Conference on Grid and Pervasive Computing, IEEE,2008.
[21] R.Buyya. Cloud Simulator cloudsim version 2.1, GRIDS Lab, http://code.google.com/p/cloudsim, July 27, 2010.