Job Classification in Cloud Computing: The Classification Effects on Energy Efficiency

Auday Al-Dulaimy *
Department of Mathematics & Computer Science,
Beirut Arab University,
Beirut, Lebanon
auday.aldulaimy@gmail.com

Rached Zantout
Department of Electrical & Computer Engineering,
Rafic Hariri University,
Beirut, Lebanon
zantoutrn@rhu.edu.lb

Ahmed Zekri **
Department of Mathematics & Computer Science,
Beirut Arab University,
Beirut, Lebanon
a.zekri@bau.edu.lb

Wassim Itani
Department of Electrical & Computer Engineering,
Beirut Arab University,
Beirut, Lebanon
w.itani@bau.edu.lb

***Abstract*— One of the recent and major challenges in cloud computing is to enhance the energy efficiency in cloud data centers. Such enhancements can be done by improving the resource allocation and management algorithms. In this paper, a model that identifies common patterns for the jobs submitted to the cloud is proposed. This model is able to predict the type of the job submitted, and accordingly, the set of users' jobs is classified into four subsets. Each subset contains jobs that have similar requirements. In addition to the jobs' common pattern and requirements, the users' history is considered in the jobs' type prediction model. The goal of job classification is to find a way to propose useful strategy that helps improve energy efficiency. Following the process of jobs' classification, the best fit virtual machine is allocated to each job. Then, the virtual machines are placed to the physical machines according to a novel strategy called Mixed Type Placement strategy. The core idea of the proposed strategy is to place virtual machines of the jobs of different types in the same physical machine whenever possible, based on Knapsack Problem. This is because different types of jobs do not intensively use the same compute or storage resources in the physical machine. This strategy reduces the number of active physical machines which leads to major reduction in the total energy consumption in the data center. A simulation of the results shows that the presented strategy outperforms both Genetic Algorithm and Round Robin from an energy efficiency perspective.**

Keywords- Cloud Computing; Data Center; Virtualization Management; Energy Efficiency.

I. INTRODUCTION

Reducing energy consumption in data centers has become essential nowadays. Such reduction should be based on power delivered to computing resources and resources utilization. In [1] and [2] dynamic voltage and frequency scaling (DVFS) is used. In [3], dynamically adjusting the number of CPUs in a cluster enabled energy saving when utilization is low. In [4] and [5] switching physical machines (PMs) off was used to save energy. In [6] and [7], energy minimization was based on statistically analyzed the workload. The studies in [8], [9] and [10] optimize consumed power, service cost, and overall performance. The authors in [11] represent service requests as a function to minimize energy consumption. In [12] tasks on heterogeneous machines are scheduled according to their energy cost to maximize profit. In [13], customer utilization patterns are proposed in a dynamic resource provision mechanism. The proposed patterns place additional Virtual Machines (VMs) on the PMs of cloud data centers where possible. The impact on the trade-off between energy efficiency and SLA requirements was analyzed. In [14], an energy-efficient job scheduling and allocation scheme is presented which is claimed to reduce the number of active PMs. In [15] the IaaS cloud service model is used as a computing infrastructure by provisioning VMs on-demand. In [16] distributed dynamic consolidation of VMs in virtualized cloud data centers is presented which is based on the bin packing algorithm.

The rapid growth in cloud computing model has emphasized the need for improving the existing resource allocation and management algorithms in cloud data centers as well as proposing new ones. In this paper, a new model is suggested to predict the job type in the workload based on its common behaviors and patterns. User history, when available, is also used in predicting the job type. According to the job type, the best fit VM is allocated to job. Then VMs are placed to the PMs using the Knapsack Problem (KP) such that no VMs having the same type of jobs will consolidate on the same PM. High Performance Computing (HPC) jobs can be effectively combined with Data Intensive (DI) jobs on the same PM. HPC jobs mostly request compute resources, whereas DI jobs request storage and network bandwidth resources. This leads to the reduction in the number of active PMs. VM placement is done by using KP. Other methods, such as bin packing algorithm [16] are available. In this paper, Multi Choice Knapsack Problem (MCKP) is used since selection is done from different sets [17]. In this paper, reducing energy consumption is done also by minimizing the CPU frequency the VM's of the DI jobs using DVFS. This is only applied when VM's of DI jobs exist in space shared policy. This is to prevent any effect on the other VMs on

* PhD candidate at the Department of Mathematics & Computer Science, Beirut Arab University, Beirut, Lebanon. Email: a.aldulaimy@student.bau.edu.lb

** On leave from the Department of Mathematics & Computer Science, Faculty of Science, Alexandria University, Alexandria, Egypt. Email: ahmed.zekri@alexu.edu.eg

978-0-7695-5697-0/15 $31.00 © 2015 IEEE
DOI 10.1109/UCC.2015.97

the PM as well as not violating the DI job's Quality of Service (QoS). In this paper, one VM is assumed to be allocated to each job. Jobs are assumed to be organized as Bag-of-Tasks (BoT), independent, executable in any order.

In section 2 the components of the system model which submit users' jobs to the cloud are presented. Jobs classification is also detailed. In section 3 the model is described along with the new MTP strategy. Experimental results are discussed and analyzed in section 4. Finally, conclusions and future works are listed in section 5.

II. SYSTEM COMPONENTS

The system is based on the cloud computing environment paradigm, whereby users request services from a Cloud provider to execute jobs. The system consists of two main components: *User* and *Provider*, as shown in Figure 1. The cloud user submits the job (or set of jobs) to the cloud provider to be executed, typically with an implied QoS requirement. The submission model is shown below:

$$job_i(QoS)$$

QoS includes the deadline (DL_i) and budget (B_i) for the user jobs. A Provider receives the users' jobs, executes them at a specific data center, then sends the results back to the users.

The Provider component includes two models: G*lobal Scheduler* and *Data Center(s)*. The global scheduler acts as an interface between users and cloud infrastructure. It profiles and analyzes the service requirements of the submitted jobs and decides whether to accept or reject them based on the availability of resources. Then, it selects the data center that should execute the jobs so that energy consumption can be reduced, while meeting QoS requirements. The global scheduler model operates in three main phases: *Prediction Phase*, *Service Level Agreement (SLA) Phase*, and *Mapping Phase*. In the prediction phase job types are decided. A job is classified as high performance computing (HPC), data intensive (DI), high performance computing and data intensive (HPC-DI), or Normal (N).

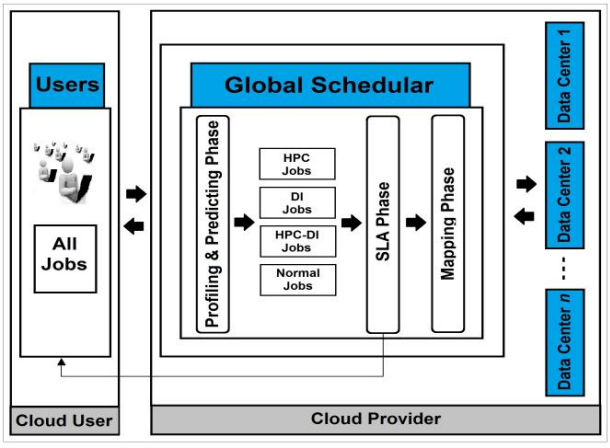

Figure 1. System Components.

Job profiling creates the following infrastructure requirements:

$(PeNumber_{User}, Time_{User}, Storage_{User}, RAM_{User}, BW_{User})$

where:

- $PeNumber_{User}$: *Number of Process Elements* (PEs) *needed* by job_i, *also known as cores.*
- $Time_{User}$: *Required execution time for* job_i *when using* $PeNumber_{User}$.
- $Storage_{User}$: *Size of the storage needed by* job_i
- RAM_{User} : *RAM size needed by* job_i
- BW_{User} : *Bandwidth needed by* job_i

Execution time is estimated in [8] and [18] as a known quantity or given by the User. Code analysis is used in [19], while Analytic benchmarking/code profiling is used in [20]. Historical/Statistical prediction is used in [21] while Empirical analysis is used in [22]. Cycle Per Instruction (CPI) was used in [23], while Memory Access Per Instruction (MPI) was used in [24], and estimated bandwidth was used in [25]. After profiling, this phase utilizes a mix of statistical information and some input features for the jobs to predict the job type. This is done in two ways. The first is the *User Id check* (UIC) A Log File (LF) is maintained for each user which is used in profiling Users as well as jobs. LF is considered as a list of recordsof the form:

Job (JobID, JobType, Date, UserID)

The second way is the *Job Features Check* (JFC). Profiling the job by examining the following features: total execution time of the job, Cycle Per Instruction (CPI), Memory Access Per Instruction (MPI), Size of the job, and Bandwidth.

A new algorithm, User Type Predictor (UTP), is used to predict the users type based on their history. The prediction is based on the type of the last number of jobs submitted or the type of jobs submitted within a past period of time, or both.

Algorithm: User Type Predictor

```
Input:    LF , nJOB, nDAY
Output:   User type
1         JobCounter=0;
2         StartDate = Current Date
3         EndDate = StartDate – nDAY
4         while ( JobCounter < nJOB ) or ( LE has more jobs )
5               If (Job.Date≤ StartDate & Job.Date≥EndDate )
6                     switch (Job.JobType)
7                           Case HPC: CounterHPC ++
8                           Case DI: CounterDI ++
9                           Case HPC-DI: CounterHPC-DI ++
10                          Case Normal: CounterNormal ++
11                    end switch
12                    JobCounter ++
13              end if
14        end while
15        return (Type of the max counter)
```

Algorithm 1: User Type Predictor Algorithm

In algorithm 1, nJOB (which represents n jobs submitted by a specific user) and nDAY (which represents n days of jobs' submission by a specific user) may vary over time according to the user's identity and the types of job submitted by this user.

Equations (1) to (4) calculate the total values requested for time execution, memory, storage, and bandwidth respectively, which are required to execute jobs for a specific user.

$$ReqTime_{user} = \sum_{i=1}^{NoOfJobs} Time_i \quad (1)$$

$$ReqMemory_{user} = \sum_{i=1}^{NoOfJobs} RAM_i \quad (2)$$

$$ReqStorage_{user} = \sum_{i=1}^{NoOfJobs} Storage_i \quad (3)$$

$$ReqBW_{user} = \sum_{i=1}^{NoOfJobs} Bw_i \quad (4)$$

Equations (5) to (8) are used to specify the type of the user.

$$HPC = (LF_{HPC} * w_1) + \Big(\big((job_i^{ET} \geq ExecTh) \; or \; (job_i^{CPI} \geq CpiTh) \; or \; (job_i^{MPI} \geq MpiTh) \big) * w_2 \Big) + \big(\big((job_i^{size} < SizeTh) \; or \; (job_i^{BW} < BwTh) \big) * w_3 \big) \quad (5)$$

$$DI = (LF_{DI} * w_1) + \Big(\big((job_i^{ET} < ExecTh) \; or \; (job_i^{CPI} < CpiTh) \; or \; (job_i^{MPI} < MpiTh) \big) * w_2 \Big) + \big(\big((job_i^{size} \geq SizeTh) \; or \; (job_i^{BW} \geq BwTh) \big) * w_3 \big) \quad (6)$$

$$HPC - DI = (LE_{BOTH} * w_1) + \Big(\big((job_i^{ET} \geq ExecTh) \; or \; (job_i^{CPI} \geq CpiTh) \; or \; (job_i^{MPI} \geq MpiTh) \big) * w_2 \Big) + \big(\big((job_i^{size} \geq SizeTh) \; or \; (job_i^{BW} \geq BwTh) \big) * w_3 \big) \quad (7)$$

$$NORMAL = (LE_{NORMAL} * w_1) + \Big(\big((job_i^{ET} \leq ExecTh) \; or \; (job_i^{CPI} \leq CpiTh) \; or \; (job_i^{MPI} \leq MpiTh) \big) * w_2 \Big) + \big(\big((job_i^{size} \leq SizeTh) \; or \; (job_i^{BW} \leq BwTh) \big) * w_3 \big) \quad (8)$$

where:

$$LF_{HPC} = \begin{cases} 1 & if\ the\ history\ of\ the\ user\ in\ LF\ is\ HPC \\ 0 & otherwise \end{cases}$$

$$LF_{DI} = \begin{cases} 1 & if\ the\ history\ of\ the\ user\ in\ LF\ is\ DI \\ 0 & otherwise \end{cases}$$

$$LF_{BOTH} = \begin{cases} 1 & if\ LF_{HPC} = LF_{DI} = 1 \\ 0 & otherwise \end{cases}$$

$$LF_{NORMAL} = \begin{cases} 1 & if\ LF_{HPC} = LF_{DI} = 0 \\ 0 & otherwise \end{cases}$$

$ExecTh$: threshold for job execution time
$CpiTh$: threshold for CPI of the job
$MpiTh$: threshold for MPI of the job
$SizeTh$: threshold for job size
$CompTh$: threshold for the required bandwidth for the job

$w1$,$w2$, and $w3$ are weight values satisfying

$w1 + w2 + w3 = 1$,

The cloud provider assigns the weight values. $w1$ is related to the User history (if any). $w2$ is related to total execution time, CPI, and MPI, values which high in HPC jobs. $w3$ is related job size and bandwidth which are high for DI jobs.

Thresholds are selected by examining real workload trace. A histogram (e.g. Figure 2 for CPI threshold) is sketched and the average and median are used to calculate the threshold.

The job type is selected according to the maximum value of resulted from (1) to (4). If two or more values are equal (ambiguity state), the type in the LF should be the dominant one. If there is no information about the user in the LF, w_1 is zero.

In the SLA phase, an agreement about the offered services between the user and cloud provider is achieved to ensure that the QoS requirements of the users are met [26]. The global scheduler decides whether or not the provider can execute the user's job if the provider is not able to execute the job with the required QoS, the user will be informed in this phase. Otherwise, the job will be passed on to the mapping phase.

In the mapping phase, mapping jobs to data centers is performed to minimize the total amount of energy consumption. This is done by interacting with the local schedulers of each data center [16] to determine CPU availability, free time slots, available, and expected amount of consumed energy when executing the jobs on the available resources of the data center. The mapping process makes sure not to violate the SLA constraints. The local scheduler performs the actual jobs scheduling.

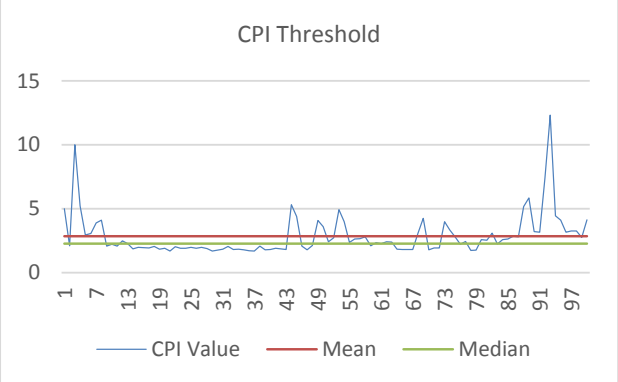

Figure 2. Selecting CPI threshold for a Set of Jobs

The data center model (presented in Figure 3) consists of n heterogeneous PMs. Each PM_i contains multicore processors and is characterized by the configuration shown below:

$$PM_i(PE_{PM}, Speed_{PM}, Storage_{PM}, RAM_{PM}, BW_{PM})$$

where:

- PE_{PM} : $Number\ of\ PEs\ in\ PM_i$
- $Speed_{PM}$: $Speed\ of\ each\ PE\ in\ PM_i$
- $Storage_{PM}$: $Size\ of\ the\ storage\ in\ PM_i$
- RAM_{PM} : $RAM\ size\ in\ PM_i$
- BW_{PM} : $Bandwidth\ in\ PM_i$

Each PM can have one or more VMs assigned. Each VM configuration is characterized as shown below:

$$VM_i(peNumber_{VM}, Speed_{VM}, Storage_{VM}, RAM_{VM}, BW_{VM})$$

where:

- $peNumber_{VM}$: $Number\ of\ PEs\ in\ VM_i$
- $Speed_{VM}$: $Speed\ of\ each\ PE\ in\ VM_i$
- $Storage_{VM}$: $Size\ of\ the\ storage\ in\ VM_i$
- RAM_{VM} : $RAM\ size\ in\ VM_i$
- BW_{VM} : $Bandwidth\ in\ VM_i$

III. THE MODEL

In cloud computing, the problem of allocating resources is NP-hard [10]. This is because jobs are to be effectively scheduled in distributed, heterogeneous, and virtualized environments. The system, S, is modeled as a four-tuple (D, PM, VM, J). D is a set of data centers; each element $D_d \in D$ represents a single data center in the system. PM is a set of physical machines in the data center; each element $PM_{pm,d} \in PM$ represents a single PM_{pm} in D_d. VM is a set of virtual machines associated with physical machines in the data center; each element $VM_{vm,pm,d} \in VM$ represents a single VM_{vm} on single PM_{pm} in data center D_d. J is a set of jobs; each element $job_j \in J$ represents a single job.

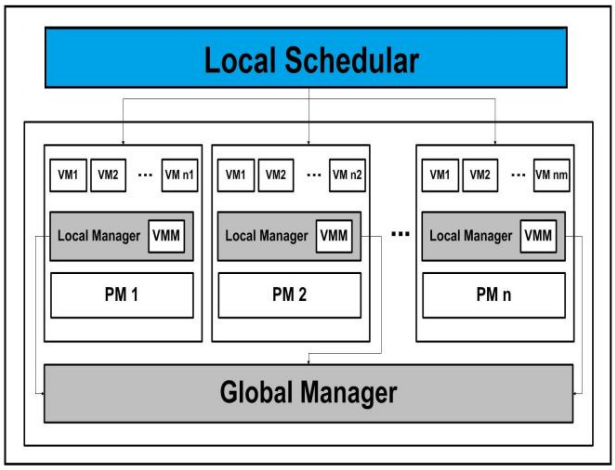

Figure 3. The Data Center Model.

Minimizing energy consumption is done by a multi-objective optimization scheduling algorithm. The energy consumed by PMs in data centers usually determined by the CPU, disk storage, memory, and network interfaces [27]. Among these components, the CPU consumes the most amount of energy. Hence, in this work, only the energy consumed by CPU using Equation (9).

$$PM_i^p = P_{CPU} + P_{Memory} + P_{Storage} \quad (9)$$

where $(PM_i^p)$is a specific PM, P_{CPU} is the power consumed by the CPU, P_{Memory} is the power consumed by memory, and $P_{Storage}$ is the power consumed by storage disk. The power consumption model of CPU is the sum of both CPU static power (P_{CPU_Static}) and CPU dynamic power $(P_{CPU\_Dynamic})$. Equation (10) is used to compute the power consumed by CPU [28]:

$$P_{CPU} = P_{CPU\_Dynamic} + P_{CPU\_Static} \quad (10)$$

where (P_{CPU_Static}) is a constant, say ω, and $P_{CPU_Dynamic}$is given in (11).

$$P_{CPU\_Dynamic} = ACV^2 f \quad (11)$$

where A is an activity factor that accounts for frequency gates switching, C is the total capacitance at the gate outputs, V is the voltage of the CPU, and f is the operating frequency. Voltage V can be expressed as a linear function of frequency, $V = \alpha f$, such that α is constant. All constants (ω, A, and C) can be combined together in one constant β. Therefore, (10) can be rewritten as shown in (12):

$$P_{CPU} = \beta f^3 \quad (12)$$

When the processor has less work, it can be slowed down without affecting performance adversely by using DVFS.

If n is the total number of jobs, the total energy consumption for executing these jobs in a data center (TP_d) can be measured using (13):

$$TP_d = \sum_{i=1}^{n} x_i * PM_i^p \quad (13)$$

where x_i is equal to 0 if the machine PM_i is off, and equal to 1 if it is on. Thus, the objective of the proposed system is to minimize the values of TP_d, as in (14).

$$Minimize\ \ TP_d = \sum_{i=1}^{n} x_i * PM_i^p \quad (14)$$

According to the job requirements, the provider performs the VM provisioning to user's job (i.e. VM allocation). There are two main policies for VMs to jobs allocation in cloud computing environments [29] that can be used. Space-shared policy which results in a VM associated with one or more cores. Time-shared policy which results in a core that holds two or more VMs. Every job is allocated to a VM with a specific frequency. When DI jobs are allocated to VMs in a space shared policy, the core frequency is minimized. This leads to better energy efficiency due to the cubic relation between energy and frequency as illustrated in (12). After VM provisioning, the

provider performs the VM placement, which is the process of placing the VM on the proper PM, using MCKP model [17]. Two VMs which belong to the same type of jobs are placed on different PM's. This leads to further reduction of the number of active PMs which is guaranteed by KP. However, when there is no available PM for combining VMs of different types of jobs, classical KP algorithm is used to place the VMs.

IV. PERFORMANCE ANALYSIS

The CloudSim simulator [30] was used to simulate a real workload trace in a cloud computing environment. The Google workload traces [31], collected from large cloud systems (about 12,500 compute nodes over 29 days) are used in the simulation. The traces consist of different types of jobs [32]. Three parameters were used in the simulations: number of PMs, number of VMs, number of jobs in the workload. The PMs and VMs configurations are as those provided by Amazon cloud data centers [33]. Table 1 summarizes the VMs instance types, called M3 types, used in the simulations. The evaluation uses the total consumed energy as a measurement. The simulations were repeatedly conducted for different numbers of PMs, VMs, and jobs. In this paper only results for 100 PMs and 400 VMs are shown. The sets of jobs tested consist of 50, 200, 400, 600, 800, and 1000 jobs. In order to evaluate the proposed MTP strategy, it was compared with Round Robin (RR) and Genetic Algorithm (GA) models. GA is a general purpose optimization technique inspired by the biological evolution. The initial population is randomly produced in the experiment. It evolves better approximate solutions from generation to generation iteratively based on specific fitness function (the consumed energy in the experiment). The individual VMs combine and cross by the genetic operators. Each new population represents a new solution of VMs to PMs placement. Similar work can be found in [34]. In RR, VMs are placed and distributed to PMs in the data center sequentially in a circular manner, (e.g. the one in Eucalyptus) which is open source software for building clouds [35]. The measurements comparison with GA and RR of the total consumed energy resulted from executing 1000 jobs are presented in Fig 4.

Table 1: VM instance types in M3 family offered by Amazon

VM Type	CPU	Clock	vCPU	Mem	BW
M3.meduim	Intel Xeon E5-2670 v2 Processors	2500	1	3750	Moderate
M3.large	Intel Xeon E5-2670 v2 Processors	2500	2	7500	Moderate
M3.xlarge	Intel Xeon E5-2670 v2 Processors	2500	3	15000	High
M3.2xlarge	Intel Xeon E5-2670 v2 Processors	2500	4	30000	High

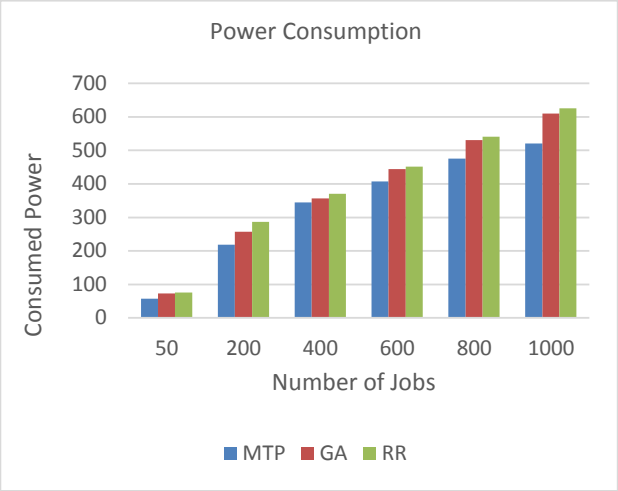

Figure 4. Consumed energy when executing different sets of jobs.

The energy consumption of each set of jobs increased as the number of jobs increased. This is because the total execution time of a job increases as more jobs are submitted. Within the same set of jobs and from energy efficiency perspective, MTP outperforms both GA and RR as illustrated in Fig 4. This is due to the fact that the compute and storage resources of each PM are optimally utilized. Such utilization prevents the idle state for the resources (which is not guaranteed in GA and RR). The total number of switch on PMs needed to execute each set of jobs is reduced.

V. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, a model that identifies common patterns for the jobs submitted to the cloud is presented. This model predicts the type of the job submitted; and accordingly, the set of users' jobs is classified into four subsets. Each subset contains jobs that have similar requirements. The goal of job classification is to find a way to propose a useful strategy that helps improve energy efficiency. Following the process of jobs' classification, a new placement strategy (MTP) is proposed. Its core concept is to place VMs of the jobs of different types in the same PM. The initial evaluation of MTP shows promising results with regard to the reduction of the total consumed energy of the data center. This is because the compute and storage resources of each PM are optimally utilized. Such utilization prevents the idle state for the resources. Consequently, the total number of PMs needed to execute the jobs is reduced.

As a future work, a performance model to explore the trade-off between energy efficiency and QoS will be developed. The effects of the proposed strategy on some of QoS factors such as time and budget will be analyzed, Also, the VM management approaches, such as VM migration and consolidation, will be applied to the proposed model to make it more integral to work in cloud computing paradigm.

REFERENCES

[1] T. Horvath, T. Abdelzaher, K. Skadron and X. Liu, "Dynamic voltage scaling in multi-tier web servers with end-to-end delay control," *IEEE TRANSACTIONS ON COMPUTERS*, vol. 56, no. 4, pp. 444-458, 2007.

[2] X. Wang, X. Fu, X. Liu and Z. Gu, "Power-aware CPU utilization control for distributed real-time systems," in *Real-Time and Embedded Technology and Applications Symposium* , San Francisco, CA, 2009.

[3] B. Lawson and E. Smirni, "Power-aware Resource Allocation in High-end Systems via Online Simulation," in *Proceedings of the 19th annual international conference on Supercomputing*, New York, NY, USA, 2005.

[4] W. Lang and J. M. Patel, "Energy management for map-reduce clusters," in *Proceedings of the 36th International Conference on Very Large Data Bases*, Singapore, 2010.

[5] J. Heo, P. Jayachandran, I. Shin, D. Wang, T. Abdelzaher and X. Liu, "OptiTuner: On Performance Composition and Server Farm Energy Minimization Application," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 11, pp. 1871-1878, 2011.

[6] D. Bradley, R. Harper and S. Hunter, "Workload-based power management for Parallel computer systems," *IBM Journal of Research and Development*, vol. 47, no. 5-6, pp. 703-718, 2003.

[7] B. Guenter, N. Jain and C. Williams, "Managing cost, performance, and reliability tradeoffs for energy-aware server provisioning," in *Proceedings IEEE INFOCOM* , Shanghai, 2011.

[8] S. K. Garg, C. S. Yeob, A. Anandasivamc and R. Buyya, "Environment-conscious scheduling of HPC applications on distributed Cloud-oriented data centers," *Jornal of Parallel Distributed Computing*, vol. 71, no. 6, pp. 732-749, 2011.

[9] G. Tesauro, R. Das, H. Chan, J. O. Kephart, C. Lefurgy, D. W. Levine and F. Rawson, "Managing power consumption and performance of computing systems using reinforcement learning," in *In Proceedings of the 21st Annual Conference on Neural Information Processing Systems*, Vancouver, Canada, 2007.

[10] F. Zhang, J. Cao, K. Li, S. U. Khan and K. Hwang, "Multi-objective scheduling of many tasks in cloud platforms," *Future Generation Computer System*, vol. 37, pp. 309-320, 2014.

[11] J. S. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat and R. P. Doyle, "Managing energy and server resources in hosting centers," in *Proceedings of the eighteenth ACM symposium on operating systems principles*, New York, NY, USA, 2001.

[12] J. Burge, P. Ranganathan and J. L. Wiener, "Cost-aware scheduling for heterogeneous enterprise machines (CASH'EM), HPL-2007- 63, ,," Hewlett-Packard Development Company, Palo Alto, 2007.

[13] R. VIKRAM and A. NEELIMA, "Resource Over allocation to Improve Energy Efficiency in Real-Time Cloud Computing Data Centers," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 2, no. 1, pp. 447-453, 2013.

[14] S. S. Deore and A. N. Patil, "Energy-Efficient Job Scheduling and Allocation Scheme for Virtual Machines in Private Clouds," *International Journal of Applied Information Systems*, vol. 5, no. 1, pp. 56-60, 2013.

[15] E. Feller, "Autonomic and Energy-Efficient Management of Large-Scale Virtualized Data Centers," PhD thesis, ISTIC, University of Rennes, 2013.

[16] A. Beloglazov, "Energy-Efficient Management of Virtual Machines in Data Centers for Cloud Computing," PhD thesis, Department of Computing and Information Systems, The University of Melbourne, 2013.

[17] D. Pisinger, "Algorithms for Knapsack Problems," PhD thesis, University of Copenhagen, 1995.

[18] G. Lee, Resource Allocation and Scheduling in Heterogeneous Cloud Environments, PhD thesis, College Of Engineering, University Of California, Berkeley, 2012.

[19] G. Nudd, D. Kerbyson, E. Papaefstathiou, S. Perry, J. Harper and D. Wilcox, "PACE: A Toolset For The Performance Prediction Of Parallel And Distributed Systems," *International Journal of High Performance Computing Application*, vol. 14, no. 3, pp. 228-251, 2000.

[20] J. Yang, I. Ahmad and A. Ghafoor, "Estimation of execution times on heterogeneous supercomputer architectures," in *International Conference on Parallel Processing*, 1993.

[21] H. Sanjay and S. Vadhiyar, "Performance Modeling Of Parallel Applications For Grid Scheduling," *Journal of Parallel Distributed Computing*, vol. 68, no. 8, pp. 1135-1145, 2008.

[22] F. Berman, H. Casanova, A. Chien, K. Cooper, H. Dail, A. Dasgupta, W. Deng, J. Dongarra, L. Johnsson, K. Kennedy, C. Koelbel, B. Liu, X. Liu, A. Mandal, G. Marin, M. Mazina, J. Crummey, C. Mendes, A. Olugbile, J. Patel, D. Reed, Z. Shi and O. Siever, "New grid scheduling and rescheduling methods in the grads project," *International Journal of Parallel Programming*, vol. 33, no. 2, pp. 209-229, 2005.

[23] J. Chen and L. K. John, "Predictive Coordination of Multiple On-Chip Resources for Chip Multiprocessors," in *International Conference on Supercomputing*, Tuscon, Arizona, USA, 2011.

[24] Z. Zhang and J. M. Chang, "A Cool Scheduler for Multi-Core Systems Exploiting Program Phases," *IEEE TRANSACTIONS ON COMPUTERS*, vol. 63, no. 5, pp. 1061-1073, 2014.

[25] J. Zhu, D. Li, J. Wu, H. Liu, Y. Zhangy and J. Zhang, "Towards bandwidth guarantee in multi-tenancy cloud computing networks," in *International Conference on Network Protocols*, Austin, TX, 2012.

[26] S. B. S. M. Moustafa, SLA Monitoring For Federated Cloud Services, MSc thesis, School of Computing, Queen's University, 2015.

[27] A. Beloglazov, J. Abawajyb and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing," *Future Generation Computer Systems*, vol. 28, no. 5, pp. 755-768, 2012.

[28] M. T. Chaudhry, T. C. Ling, A. Manzoor, S. A. Hussain and J. Kim, "Thermal-Aware Scheduling in Green Data Centers," *ACM Computing Surveys*, vol. 47, no. 3, p. Article 39, 2015.

[29] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. D. Rose and R. Buyya, "CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms," *Software: Practice and Experience Journal*, vol. 41, no. 1, pp. 23-50, 2011.

[30] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. D. Rose and R. Buyya, "Cloudsim: A Toolkit For Modeling And Simulation Of Cloud Computing Environments And Evaluation Of Resource Provisioning Algorithms," *Software—Practice & Experience Journal*, vol. 41, no. 1, pp. 23-50, 2011.

[31] Google, "https://cloud.google.com/storage/docs/overview," [Online], 2015.

[32] J. W. Charles Reiss, "Google Cluster-Usage Traces: Format And Schema," Google Inc, version 2, 2013.

[33] Amazon, "http://aws.amazon.com," [Online], 2015.

[34] Y.-S. Dong, G.-C. Xu and X.-D. Fu, "A Distributed Parallel Genetic Algorithm of Placement Strategy for Virtual Machines Deployment on Cloud Platform," *The Scientific World Journal*, vol. 2014, no. Article ID 259139, 2014.

[35] Eucalyptus, "https://www.eucalyptus.com," [Online], 2015.