Job Submission in the Cloud: Energy Aware Approaches

Auday Al-Dulaimy, Rached Zantout, Wassim Itani, and Ahmed Zekri

***Abstract*- Cloud computing model offers various types of services. These services are delivered by providing an access to a wide range of shared resources which are hosted in cloud data centers. Data centers consume large amounts of energy. One of the recent challenges in this model is to enhance the energy efficiency in these data centers. This paper tackles the problem of enhancing the energy consumption of cloud data centers by proposing three energy aware approaches. Each approach includes its innovative job classification model and its novel VM placement strategy. The job classification model classifies cloud users' jobs before serving these jobs. The model identifies common patterns for the submitted jobs and predicts their types, and accordingly, the set of users' jobs is classified into 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. Based on the process of jobs' classification, the best fit virtual machine is allocated to each job. Then, the VM placement strategy place the VM on a selected physical machine. The goal of the proposed placement strategy is to better utilize the involved physical machines which serve the users' jobs. As different types of jobs do not intensively use the compute and/or non-compute resources in the hosted physical machine, virtual machines allocated to the jobs of different types are placed on the same physical machine where possible. The placement process is based on a Multiple Choice Knapsack Problem which is a generalization of the classical Knapsack Problem. The proposed approaches enhance the energy efficiency of the cloud data centers by minimizing the number of the involved physical machines which serve the jobs, and by optimally utilizing the involved physical machines. To evaluate the performance of the proposed approaches and compare which is better, the CloudSim simulator is used with a real workload trace to simulate the cloud computing environment.**

***Index Terms*— Cloud Computing, Job Submission, Job Classification, VM Placement, Energy Efficiency.**

Manuscript received July 19, 2016; revised August 16, 2016.

Auday Al-Dulaimy: Department of Mathematics & Computer Science, Beirut Arab University, Beirut, Lebanon. (Email: auday.aldulaimy@gmail.com ; a.aldulaimy@student.bau.edu.lb).
Rached Zantout: Department of Electrical & Computer Engineering, Rafic Hariri University, Beirut, Lebanon. (Email: zantoutrn@rhu.edu.lb).
Wassim Itani: Department of Electrical & Computer Engineering, Beirut Arab University, Beirut, Lebanon. (Email: w.itani@bau.edu.lb).
Ahmed Zekri: Department of Mathematics & Computer Science, Beirut Arab University, Beirut, Lebanon. (Email: a.zekri@bau.edu.lb), on leave from Department of Mathematics & Computer Science, Alexandria University, Alexandria, Egypt. (Email: ahmed.zekri@alexu.edu.eg).

I. INTRODUCTION

Cloud computing is the computing model that builds on top of several predecessor models, called cluster computing, distributed computing, autonomic computing, utility computing, and can be considered as a step forward from the grid computing model. Cloud computing leverages these existing models, together with the virtualization concept, to meet the cloud users' requirements with new features. This model provides an access to a wide range of shared hardware resources existing in data centers around the world. These resources represent the hardware layer of the cloud computing model. Cloud users do not interact with the hardware layer directly. Instead, they benefit from such resources via a virtualization layer. Virtualization is creating a virtual version of a real thing. Virtualization is the key concept of cloud model that hides the details of physical hardware and provides virtualized resources for user layer. A single Physical Machine (PM), which is the real hardware, can host one Virtual Machine (VM) or more. A VM is a piece of software running on PM. It simulates the properties of a separated PM. VM Management is the process of coordinated provisioning of the virtualized resources. This feature includes mapping virtual resources to physical resources in addition to overall management capabilities. Recently, cloud computing has grown rapidly which applies that there is a need for establishing numerous data centers. This growth concurs with some challenges. One of the recent challenges in cloud computing is to enhance the energy efficiency of such data centers which contain thousands of compute nodes. Therefore, it is crucial to apply energy-efficient approaches when serving users' jobs in the resources of the cloud data centers.

Many previous approaches have been suggested to solve the problem of VM placement. The basic and simplest VM placement method is Round Robin, where VMs are placed on PMs sequentially. However, there are more advanced approaches. The work in [1] relied on the linear and quadratic programming in proposing a VM placement algorithm to minimize the number of the involved PMs. The authors in [2] proposed a framework to maximize the utilization of the VMs resources. They defined a dynamic VM provisioning manager and a dynamic VM placement manager, to work together within the proposed framework. VM provisioning and VM placement are modeled as constraint satisfaction problems. In [3], the authors proposed a resource manager, called Entropy, working in homogeneous clusters. Entropy performed dynamic consolidation based on constraint programming. Many other studies presented the VM placement problem as a bin packing problem. The thesis in [4] is an example of using bin packing approach in VM placement. It presented novel models and algorithms for distributed dynamic consolidation of VMs in cloud data centers. The applied a VM placement algorithm in the thesis was based on a variant of bin packing called best fit decreasing bin packing. In addition to the previous approaches, various optimization methods such as, Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), and Genetic Algorithms (GA) were used to solve the problems of VM placement, and VM consolidation, in the virtualized data centers. In [5], the authors defined an initial VM placement strategy with a multi-objective optimization algorithm based on (ACO). The proposed algorithm was able to achieve an optimal solution through efficient convergence by the constantly updated pheromone. The optimal solution was selected from a set of solutions using the exclusion method. In [6], the authors designed a distributed ACO-based algorithm for solving the VM consolidation problem. The algorithm iterated over a finite number of cycles. At each cycle, an item is selected for each ant to be packed in a bin. In [7], the authors proposed a multi-objective ACO to solve the problem of VM placement. The formulated multi-objective VM placement problem represented permutation of VM assignments. The goal was to efficiently obtain a set of non-dominated solutions that reduce the total high power consumption resulting from resource wastage. The thesis in [8] used an ACO-based VM placement algorithm to solve the problem of considering only a single resource to evaluate the PM load, and VM resource demands, while ignoring the other resources. A consolidation algorithm based on ACO to achieve both scalability, and high data center utilization, was proposed also in this thesis. In [9], the authors proposed a VM consolidation scheme that focused on balanced resource utilization of servers across different computing resources. The work in [10] proposed a PSO based VM scheduling strategy for VM placement. The strategy focused on efficient VM placement, aiming to minimize the number of the PMs used. In [11], the authors proposed a model, based on PSO, to place the migrated VMs in the over-loaded PMs on other hosts, and to consolidate the under-loaded PMs in order to save power. An example of GA approach was presented in [12] by proposing a genetic algorithm for power-aware (GAPA) scheduling to find the optimal solution for the problem of VM allocation. In the proposed algorithm, a tree structure was used to encode chromosome of an individual job. The fitness function of GA is calculating the evaluation value of each chromosome. Using this model, each instance of the tree structure showed the VM to PM Allocation.

This work tackles the problem of high energy consumption in cloud data centers by proposing three energy efficiency job submission approaches. Each has its new job classification model, and its VM placement strategy which employ the jobs' subsets resulted from its corresponding classification model.

The rest of this paper is organized as follows: Section 2 describes the proposed job submission models in details. The performance analysis of the proposed approaches are presented in section 3. The conclusions are listed in section 4.

II. JOB SUBMISSION APPROACHES

This section presents three job submission approaches: CI-DI approach, 16-10 approach and 16-8 approach. Each approach has its innovative jobs classification model which aims to classify the jobs summited to the cloud into new subsets. Every resulted subset includes a specific type of jobs. Then, the approaches present novel strategies to place the VMs allocated to the submitted jobs on their best PMS. The strategies study and discuss the effects of combining the VMs allocated on the same PM to the resulted jobs from the energy efficiency perspective.

A. System Model

The proposed system relies on the cloud computing model, whereby cloud users request the services offered by the cloud providers. Thus, the two main system components are: *Cloud user*, and C*loud provider*, as shown in Fig. 1.

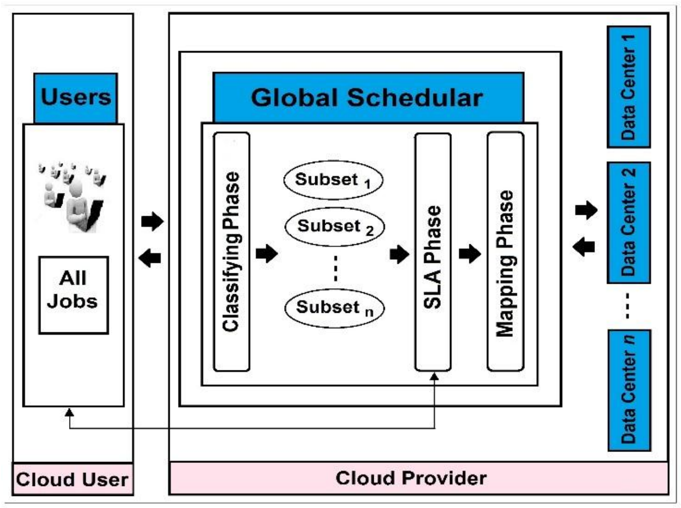

Fig. 1: System Model.

The user(s) can send their job(s) to the cloud provider. The cloud provider has an essential node called Global Scheduler (GS). This node acts as an interface between users and the cloud infrastructure and works in three phases. In the first phase (classifying Phase), the GS classifies the jobs into subsets and analyzes the service requirements of the submitted jobs. In the second phase (SLA Phase), GS decides whether to accept or reject them based on the availability of resources. In the third phase (Mapping Phase), GS selects the data center that execute the jobs such that the energy consumption can be reduced.

Data centers are located in different geographical regions. Each data center is responsible for periodically updating the GS by information about the available resources in order to achieve an energy-efficient scheduling.

Jobs are submitted as one set to the provider, every job can be considered as a finer abstraction to be served by the VM. Provider receives and classifies the jobs before sending them to be serves in a selected data center. In the data center site, there is an essential node called Local Scheduler (LS) which allocates the best fit VM configuration to each job to meet the job requirement. Then, LS performs the VM placement strategy considering the jobs type in the placement process. VM placement is the process of mapping the VMs into their best fit PMs. The VM placement process in the three proposed approaches aims to enhance the PMs utilizations in the data center which is consequently improve the energy efficiency. The VM placement process is implemented by applying one generalization on the well-known Knapsack Problem (KP). The generalization is called Multiple Choice Knapsack Problem (MCKP) [13]. It is chosen because it able to target two goals in one shot: It guarantees the minimum possible number of the involved knapsacks (in our case, every knapsack represents one PM) which minimize the consumed energy, and at the same time guarantees an acceptable utilization for the involved PMs, resulting in combining VMs which serve different types of jobs on the same PM whenever possible.

B. CI-DI Approach

In cloud computing model, VMs which serve different kind of jobs can share the same PM in the virtualized data centers. If the VMs which allocated to these jobs are placed on PMs unconditionally, then it is not obvious how these jobs influence each other in terms of performance, as they can be compute, data, or network intensive creating variable or static load on the data center resources. As stated in [4], the main reason of energy inefficiency in cloud data centers is the low utilization of the resources. So, enhancing the PMs utilization can improve the energy efficiency in cloud data centers. Enhancing the PMs utilization can be achieved by choosing the jobs that do not intensively use the same resource. A compute intensive jobs can be effectively combined with a storage and/or bandwidth intensive jobs as the former mostly relies on the compute resources, whereas the latter utilizes disk storage and/or network bandwidth. Therefore, and before the processes of VMs allocation and placements, it is important to investigate cloud workloads in order to identify common behaviors and patterns that can potentially lead to more efficient resource provisioning and consequently higher energy efficiency

In this approach, called Compute-Intensive Data-Intensive (CI-DI) approach, the set of jobs are divided into four subsets:

1) Compute Intensive (CI) jobs, which highly utilize compute resources.
2) Data Intensive (DI) jobs, which highly utilize storage and/or bandwidth resources.
3) Compute-Intensive Data-Intensive (CIDI) jobs, which utilize compute resources together with storage and/or bandwidth resources all in a high manner.
4) Normal jobs, they are the set of jobs that do not fit in any of the types specified above.

During the classification phase, a set of parameters is generated with each job as features for jobs. Some of them are: Cycle Per Instructions (CPIs), Memory Access Per Instruction (MPIs), Hard Disk (HD) usage, and Bandwidth (BW) usage. The classification is based on the four features associated with each job, the subset are generated according to the equations (1 to 4) as follows:

$$CI = ((job_i^{CPI} \geq CpiTh) or (job_i^{MPI} \geq MpiTh) + (job_i^{Size} < SizeTh) or (job_i^{BW} < BwTh)) \quad (1)$$

$$DI = ((job_i^{CPI} < CpiTh) or (job_i^{MPI} < MpiTh) + (job_i^{Size} \geq SizeTh) or (job_i^{BW} \geq BwTh)) \quad (2)$$

$$CI - DI = ((job_i^{CPI} \geq CpiTh) or (job_i^{MPI} \geq MpiTh) + (job_i^{Size} \geq SizeTh) or (job_i^{BW} \geq BwTh)) \quad (3)$$

$$NORMAL = ((job_i^{CPI} < CpiTh) or (job_i^{MPI} < MpiTh) + (job_i^{Size} < SizeTh) or (job_i^{BW} < BwTh)) \quad (4)$$

Where
- $CpiTh$: $threshold\ for\ CPI\ of\ the\ job$
- $MpiTh$: $threshold\ for\ MPI\ of\ the\ job$
- $SizeTh$: $threshold\ for\ job\ size\ to\ be\ stored\ on\ HD$
- $BwTh$: $threshold\ for\ the\ required\ bandwidth\ for\ the\ job$

The thresholds are chosen by examining the parameters' values of the all submitted jobs. Based on these values, the median of each parameter is considered as a threshold. Notably, fixed values of the threshold are not suitable for the environments with dynamic and unpredictable workloads as in cloud. So, the thresholds are dynamic in this work and changes based on the submitted jobs.

C. 16-10 Approach

The aim of this approach is to study the effects of placing the jobs that intensively utilize specific VM parameters. The core idea of this approach is to subdivide the CI and DI subsets based on their features and rely on the new sub-subsets in the process of VM placement. The CI subset can be divided into two main categories: CPU intensive and RAM intensive. Similarly, the DI subset can be divided into two main categories: Hard disk intensive and Bandwidth intensive. Further investigation to the CI and DI subset leads us to subdivide such jobs into the following sub subsets:

1) *CI jobs*: The major resources consumed by CI jobs are CPU cores and Memory. So, the CI jobs can be further subdivided into:
   a. CPU Compute Intensive (CCI) jobs: CCI jobs devote most of their execution time in utilizing the CPU cores.
   b. RAM Compute Intensive (RCI) jobs: RCI jobs devote most of their execution time in utilizing the RAM.
2) *DI jobs*: The major resources consumed by DI jobs are Hard disk and Bandwidth. So, the DI jobs can be further subdivided into:
   a. Hard disk Data Intensive (HDI) jobs: HDI jobs process large volumes of data and devote most of their execution time in the movement and manipulation of data in databases or files.
   b. Bandwidth Data Intensive (BDI) jobs: BDI jobs usually generate a huge amount of network transactions between user and cloud environment. The number of user requests and the data size of each request can highly impact the system performance and consequently the energy consumption.

The process of job classification in this approach is based on two features associated with CI jobs, and two features associated with DI jobs. The features associated with CI jobs are CPU usage and Memory usage. The features associated with DI jobs are Disk usage and Bandwidth usage.

Then, to cover all possible VM placement possibilities, the VMs allocated to the 16 possible sub-subsets based on the above four subsets are placed on PMs by applying MCKP for every combinations from the cases illustrated in Table 1.

Table 1: All possible job type combinations based on 16-10 approach.

1	CCI – CCI	Based on the usage of CPU, two sets are resulted from this classification: CCI and NOT CCI.
2	CCI – RCI	Based on the usage of CPU and RAM, four sets are resulted from this classification: CCI, RCI, CCI-RCI, and NORMAL.
3	CCI – HDI	Based on the usage of CPU and HD, four sets are resulted from this classification: CCI, HDI, CCI-HDI, and NORMAL.
4	CCI – BDI	Based on the usage of CPU and BW, four sets are resulted from this classification: CCI, HDI, CCI-HDI, and NORMAL.
5	RCI – RCI	Based on the usage of RAM, two sets are resulted from this classification: RCI and NOT RCI.
6	RCI – CCI	Based on the usage of RAM and CPU, four sets are resulted from this classification: CCI, RCI, CCI-RCI, and NORMAL.
7	RCI – HDI	Based on the usage of RAM and HD, four sets are resulted from this classification: RCI, HDI, RCI-HDI, and NORMAL.
8	RCI - BDI	Based on the usage of RAM and BW, four sets are resulted from this classification: RCI, BDI, RCI-BDI, and NORMAL.
9	HDI - HDI	Based on the usage of HD, two sets are resulted from this classification: HDI and NOT HDI.
10	HDI - BDI	Based on the usage of HD and BW, four sets are resulted from this classification: HDI, BDI, HDI-BDI, and NORMAL.
11	HDI- CCI	Based on the usage of HD and CPU, four sets are resulted from this classification: CCI, HDI, CCI-HDI, and NORMAL.
12	HDI - RCI	Based on the usage of HD and RAM, four sets are resulted from this classification: RCI, HDI, RCI-HDI, and NORMAL.
13	BDI – BDI	Based on the usage of BW, two sets are resulted from this classification: BDI and NOT BDI.
14	BDI – HDI	Based on the usage of BW and HD, four sets are resulted from this classification: HDI, BDI, HDI-BDI, and NORMAL.
15	BDI – CCI	Based on the usage of BW and CPU, four sets are resulted from this classification: CCI, HDI, CCI-HDI, and NORMAL.
16	BDI – RCI	Based on the usage of BW and RAM, four sets are resulted from this classification: RCI, BDI, RCI-BDI, and NORMAL.

From the above 16 possible combination cases, it is obvious that some of them are similar to other combinations, so repeating them does not make sense. The cases (2, 6), (3, 11), (4, 15), (7, 12), (8, 16), and (10, 14) are similar, so each pair is combined in one combination, and the 16 job type combinations is reduced to only 10 job type combinations. So, this approach is called 16-10 approach.

The VMs which are allocated to the jobs of the following 10 combinations cases can be placed together whenever possible, and the results are examined from the energy perspective. The resulted 10 combinations are: CCI *with* NOT CCI, CCI *with* RCI, CCI *with* HDI, CCI *with* BDI, RCI *with* NOT RCI, RCI *with* HDI, RCI *with* BDI, HDI *with* NOT HDI, HDI *with* BDI, and BDI *with* NOT BDI.

The 16-10 approach works with the following steps:

Step One: Divide the set of submitted jobs into 4 subsets (CCI, RCI, HDI, and BDI) as follows:

CCI job =: $CPI_i > CpiTh$
RCI job =: $MPI_i > MpiTh$
BDI job =: $HD_i > SizeTh$
CCI job =: $BW_i > BwTh$

Step Two: Try all their possible 10 combinations.

1st Combination:
- CCI job =: $CPI_i >= CpiTh$
- Not CCI job =: $CPI_i < CpiTh$
- Place VMs of CCI jobs with VMs of Not CCI jobs on the same PM whenever possible.

2nd Combination:
- CCI job =: ($CPI_i > CpiTh$) and ($MPI_i < MpiTh$)
- RCI job =: ($MPI_i > MpiTh$) and ($CPI_i < CpiTh$)
- CCI-RCI =: ($CPI_i > CpiTh$) and (MPI_i $MpiTh$)
- Normal =: ($CPI_i < CpiTh$) and ($MPI_i < MpiTh$)
- Place VMs of CCI jobs with VMs of RCI jobs, and VMs of CCI-RCI jobs with VMs of Normal jobs.

3rd Combination:
- CCI job =: ($CPI_i > CpiTh$) and ($HD_i < SizeTh$)
- HDI job =: ($HD_i > SizeTh$) and ($CPI_i < CpiTh$)
- CCI-HDI =: ($CPI_i > CpiTh$) and ($HD_i > SizeTh$)
- Normal =: ($CPI_i < CpiTh$) and ($HD_i < SizeTh$)
- Place VMs of CCI jobs with VMs of HDI jobs, and VMs of CCI-HDI jobs with VMs of Normal jobs.

4th Combination:
- CCI job =: ($CPI_i > CpiTh$) and ($BW_i < BwTh$)
- BDI job =: ($BW_i > BwTh$) and ($CPI_i < CpiTh$)
- CCI-HDI =: ($CPI_i > CpiTh$) and ($BW_i > BwTh$)
- Normal =: ($CPI_i < CpiTh$) and ($BW_i < BwTh$)
- Place VMs of CCI jobs with VMs of BDI jobs, and VMs of CCI-BDI jobs with VMs of Normal jobs.

5th Combination:
- RCI job =: $MPI_i >= MpiTh$
- Not RCI job =: $MPI_i < MpiTh$
- Place VMs of RCI jobs with Not RCI jobs.

6th Combination:
- RCI job =: ($MPI_i > MpiTh$) and ($HD_i < SizeTh$)
- HDI job =: ($HD_i > SizeTh$) and ($MPI_i < MpiTh$)
- RCI-HDI =: ($MPI_i > MpiTh$) and ($HD_i > SizeTh$)
- Normal =: ($MPI_i < MpiTh$) and ($HD_i < SizeTh$)
- Place VMs of RCI jobs with VMs of HDI jobs, and VMs of RCI-HDI jobs with VMs of Normal jobs

7th Combination:
- RCI job =: ($MPI_i > MpiTh$) and ($BW_i < BwTh$)
- BDI job =: ($BW_i > BwTh$) and ($MPI_i < MpiTh$)
- RCI-HDI =: ($MPI_i > MpiTh$) and ($BW_i > BwTh$)
- Normal =: ($MPI_i < MpiTh$) and ($BW_i < BwTh$)
- Place VMs of RCI jobs with VMs of BDI jobs, and VMs of RCI-BDI jobs with VMs of Normal jobs.

8th Combination:
- HDI job =: $HD_i >= SizeTh$
- Not HD job =: $HD_i < SizeTh$
- Place VMs of HDI jobs with Not HDI jobs.

9th Combination:
- HDI job =: ($HD_i > SizeTh$) and ($BW_i < BwTh$)
- BDI job =: ($BW_i > BwTh$) and ($HD_i < SizeTh$)
- HDI-BDI =: ($HD_i > SizeTh$) and ($BW_i > BwTh$)
- Normal =: ($HD_i < SizeTh$) and ($BW_i < BwTh$)

- Place VMs of HDI jobs with VMs of BDI jobs, and VMs of HDI-BDI jobs with VMs of Normal jobs.

10th Combination:

- BDI job =: BW$_i$ >= $BwTh$
- Not BDI job =: BW$_i$ < $BwTh$
- Place VMs of BDI jobs with VMs of Not BDI jobs

Step Three: Compute the resulted consumed energy in each combination.

D. 16-8 Approach

The aim of this approach is to consider all the possible subset of jobs which are request the VM parameter resources before the process of VM placement. The core idea of this approach is to generate 16 subset of jobs by considering the 4 VM configuration parameters (CPU, RAM, HD, and BW). This approach employs some jobs' features (CPI, MPI, HD usage, BW usage) as an indication to the resources they utilize. If the job feature value is greater than the threshold, then it is considered as a highly utilizer for the requested resource. The resulted subsets and their complement jobs are explained and listed in Table 2.

Table 2: All possible job type combinations based on 16-8 approach.

VM Parameters				**Job Type**	**Indexed as**	**Placed with**
CPU	**RAM**	**HD**	**BW**			
1	0	0	0	CPU intensive	1	16
1	0	0	1	CPU-BW intensive	2	15
1	0	1	0	CPU-HD intensive	3	14
1	0	1	1	CPU-HD-BW intensive	4	13
1	1	0	0	CPU-RAM intensive	5	12
1	1	0	1	CPU-RAM-BW intensive	6	11
1	1	1	0	CPU-RAM-HD intensive	7	10
1	1	1	1	CPU-RAM-HD-BW intensive	8	9
0	0	0	0	Normal	9	8
0	0	0	1	BW intensive	10	7
0	0	1	0	HD intensive	11	6
0	0	1	1	HD-BW intensive	12	5
0	1	0	0	RAM intensive	13	4
0	1	0	1	RAM-BW intensive	14	3
0	1	1	0	RAM-HD intensive	15	2
0	1	1	1	RAM-HD-BW intensive	16	1

The 16 job types result only 8 job type combinations. So, this approach is called 16-8 approach. Then, the VMs which are allocated to the jobs that request the complement resources are placed together whenever possible. The results after the process of VM placement can be examined from the energy perspective. As seen from Table 2, 8 possible combination are resulted.

The 16-8 approach works with the following steps:

Step one: Divide the input set of jobs into 16 subsets (CPU intensive, CPU-BW intensive, CPU-HD intensive, CPU-HD-BW intensive, CPU-RAM intensive, CPU-RAM-BW intensive, CPU-RAM-HD intensive, CPU-RAM-HD-BW intensive, Normal, BW intensive, HD intensive , HD-BW intensive, RAM intensive, RAM-BW intensive, RAM-HD intensive, RAM-HD-BW intensive).

- CPU Intensive job =: (CPI$_i$ > $CpiTh$) and (MPI$_i$ < MPI-threshold) and (HD$_i$ < HD-threshold) and (BW$_i$ < $BwTh$)
- CPU-BW intensive job := (CPI$_i$ > $CpiTh$) and (MPI$_i$ < $MpiTh$) and (HD$_i$ < HD-threshold) and (BW$_i$ > $BwTh$)
- CPU-HD intensive job := (CPI$_i$ > $CpiTh$) and (MPI$_i$ < $MpiTh$) and (HD$_i$ > $SizeTh$) and (BW$_i$ < $BwTh$)
- CPU-HD-BW intensive job =: (CPI$_i$ > $CpiTh$) and (MPI$_i$ < $MpiTh$) and (HD$_i$ > $SizeTh$) and (BW$_i$ > $BwTh$)
- CPU-RAM intensive job =: (CPI$_i$ > $CpiTh$) and (MPI$_i$ > $MpiTh$) and (HD$_i$ < $SizeTh$) and (BW$_i$ < $BwTh$)
- CPU-RAM-BW intensive job =: (CPI$_i$ $CpiTh$) and (MPI$_i$ > $MpiTh$) and (HD$_i$ < $SizeTh$) and (BW$_i$ > $BwTh$)
- CPU-RAM-HD intensive job =: (CPI$_i$ > $CpiTh$) and (MPI$_i$ > $MpiTh$) and (HD$_i$ > $SizeTh$) and (BW$_i$ < $BwTh$)
- CPU-RAM-HD-BW intensive job =: (CPI$_i$ > $CpiTh$) and (MPI$_i$ > $MpiTh$) and (HD$_i$ > $SizeTh$) and (BW$_i$ > $BwTh$)
- NONE intensive job =: (CPI$_i$ < $CpiTh$) and (MPI$_i$ < $MpiTh$) and (HD$_i$ < $SizeTh$) and (BW$_i$ < $BwTh$)
- BW intensive job =: (CPI$_i$ < $CpiTh$) and (MPI$_i$ < $MpiTh$) and (HD$_i$ < $SizeTh$) and (BW$_i$ > $BwTh$)
- HD intensive job =: (CPI$_i$ < $CpiTh$) and (MPI$_i$ < $MpiTh$) and (HD$_i$ > $SizeTh$) and (BW$_i$ < $BwTh$)
- HD-BW intensive job =: (CPI$_i$ < $CpiTh$) and (MPI$_i$ < $MpiTh$) and (HD$_i$ > $SizeTh$) and (BW$_i$ > $BwTh$)
- RAM intensive job =: (CPI$_i$ < $CpiTh$) and (MPI$_i$ > $MpiTh$) and (HD$_i$ < $SizeTh$) and (BW$_i$ < $BwTh$)
- RAM-BW intensive job =: (CPI$_i$ < $CpiTh$) and (MPI$_i$ > $MpiTh$) and (HD$_i$ < $SizeTh$) and (BW$_i$ > $BwTh$)
- RAM-HD intensive job =: (CPI$_i$ < $CpiTh$) and (MPI$_i$ > $MpiTh$) and (HD$_i$ > $SizeTh$) and (BW$_i$ < $BwTh$)
- RAM-HD-BW Intensive job =: (CPI$_i$ < $CpiTh$) and (MPI$_i$ > $MpiTh$) and (HD$_i$ > $SizeTh$) and (BW$_i$ > $BwTh$)

Step two: Try the following combinations in the process of the VM placement, and calculate the resulted consumed energy:

Place the VMs of CPU Intensive jobs *with* VMs of RAM-HD-BW Intensive jobs, CPU-BW Intensive *with* RAM-HD Intensive, CPU-HD Intensive *with* RAM-BW Intensive, CPU-HD-BW Intensive *with* RAM Intensive, CPU-RAM Intensive *with* HD-BW Intensive, CPU-RAM-BW Intensive *with* HD Intensive, CPU-RAM-HD Intensive *with* BW Intensive, CPU-RAM-HD-BW Intensive *with* NONE.

III. PERFORMANCE ANALYSIS

This section presents the evaluation of the proposed approaches. The CloudSim simulator [14], which is an extensible simulation toolkit that enables modeling and simulation of cloud computing systems and application provisioning environments, is used in the evaluation. The CloudSim supports both system and behavior modeling of cloud system components. A real workload trace to simulate the cloud computing environment is used. The jobs information is based on real data provided by Google to provide a very high level of realism when used directly in performance evaluation experiments. More details about this data are available in [15]. The PMs and VMs configurations are as those provided by Amazon cloud data centers [16]. The evaluation in this section is based on the total consumed energy as a measurement. The experiments are done and repeated 10 times for 1600 jobs in two scenarios. In the first scenario, the jobs are selected according to their types as they in the real workload trace. In the second scenario, the jobs are selected with equal distribution (i.e. 100 jobs for each type). The total consumed energy resulted in executing the jobs by applying the proposed approaches are summarized in Fig. 2 for the first scenario, and in Fig. 3 for the second scenario.

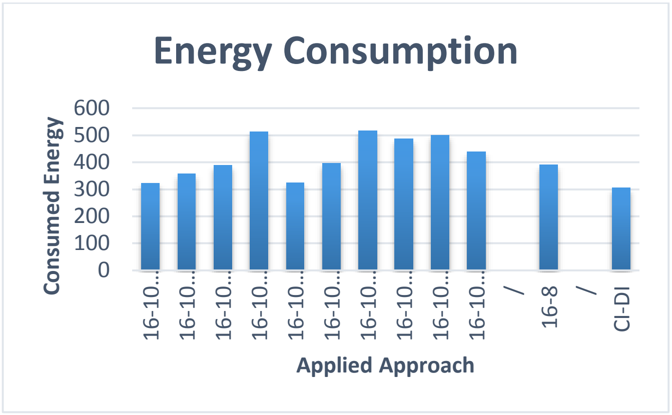

Fig. 2: Energy consumption resulted in executing 1600 jobs by applying the proposed approaches with normal distributed 16 subsets.

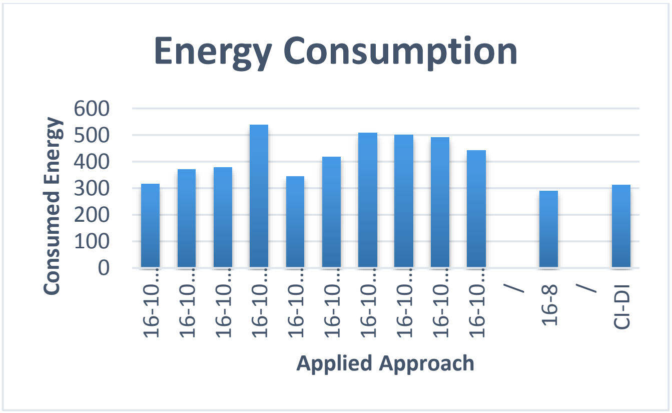

Fig. 3: Energy consumption resulted in executing 1600 jobs by applying the proposed approaches with equal distributed 16 subsets.

Applying the CI-DI approach, is better in enhancing energy efficiency because it involves minimum number of PMs in executing the set of jobs. Minimizing the number of involved PMs is a very important factor in enhancing the energy efficiency in cloud data centers, because even the completely idle PMs consume about 70% of their peak power. Another key and important reason is that the resources of the involved PMs are better utilized.

By applying the 16-10 approach (which is actually divided into 10 sub-approaches), the resulted consumed energy is vary from combination to another. In CCI-CCI combination, the resulted consumed energy is acceptable because it combines the VMs of the jobs which utilize CPU intensively with the VMs of the jobs which do not utilize CPU intensively. The CPU intensive jobs are already RAM intensive jobs, so their VMs are combined with the VMs of HDI or VMs of BDI. In RCI-RCI combination, the resulted consumed energy is also acceptable because it combines the VMs of the jobs which utilize RAM intensively with the VMs of the jobs which do not utilize RAM intensively. The RAM intensive jobs are already CPU intensive jobs, so their VMs are combined with the VMs of HDI or VMs of BDI. In HDI-HDI combination, the result is worse than CCI-CCI and RCI-RCI combinations from the energy consumption perspective. It combines the VMs of the jobs which utilize HD intensively with the VMs of the jobs which do not utilize HD intensively. The VMs of HDI jobs are combined with the VMs of CCI, RCI, BDI, or their combinations. However, combining VMs of HDI jobs with the VMs of CI jobs is satisfactory, but with the VMs of BDI, the system performance is effected because one job of them is influence the other execution by requesting the same resource. Eventually, this affect the energy efficiency. Also, in BDI-BDI combination, the result is also worse than CCI-CCI and RCI-RCI combinations from the energy consumption perspective. It combines the VMs of the jobs which utilize BW intensively with the VMs of the jobs which do not utilize HD intensively. The VMs of BDI jobs are combined with the VMs of CCI, RCI, HDI, or their combinations. However, combining VMs of BDI jobs with the VMs of CI jobs is fine, but with the VMs of HDI, the system performance is affected because one job of them is influence the other execution by requesting the same resource. Eventually, this affect the energy efficiency. In CCI-CCI, RCI-RCI, HDI-HDI, and BDI-BDI combinations, the number of the elements of the subsets are equally distributed between the subset and its complement. This is an advantage because every job has a corresponding one to combine with. The equality in the number of elements in each subset comes from using the median in calculating the thresholds. In CCI-RCI combination, the resulted consumed energy is more than CCI-CCI, RCI- RCI, HDI-HDI, and BDI-BDI combinations. In this combination, although the number of elements is small for both CCI and RCI jobs, the number of elements in CCI-RCI and Normal jobs are large and distributed almost fairly between them. So combining CCI-RCI and Normal jobs in this combination gives acceptable results since they utilize different types of resources, and every VM has can find another VM to be placed with in the same PM. In HDI-BDI combination, the resulted consumed energy is more than CCI-CCI, RCI- RCI, HDI-HDI, and BDI-BDI combinations. In this combination, although the number of elements is small for both HCI and DCI jobs, the number of elements in CCI-RCI and Normal jobs are large and distributed almost between them, but in less percentage as the elements are distributed in CCI-RCI. So combining CCI-RCI and Normal jobs in this combination gives acceptable results since they utilize different types of resources, and every VM has can find another VM to be placed with in the same PM. It is clearly seen that the energy consumption is increased when considering the CCI-HDI, RCI-BDI, CCI-HDI, and RCI-BDI combinations in the VM placement process. This is because two main reasons: the first reason is: Most of the VMs which enforced to be placed together on the same host PM are utilized the same type of resources. This decreases the PM utilization and consequently badly affects energy efficiency. The second reason is the jobs are not distributed equally between the subsets to combine the VMs of their jobs. This results in a large number of VMs which do not a VM to place with it together on the same PM. These VMs need extra computations to find their best PMs. However, the outcome that can be concluded from the experiments results of the 16-10 approach is that energy efficiency can be affected by the type of the resource which is utilized by the VM allocated to a specific job, and by the VM placement strategy which decides the VMs to be combined on the same PM based on the required resources.

By applying the 16-8 approach in the VM placement process, it is clearly seen that the energy consumption is increased dramatically. Although 16-8 approach investigates all the VM configuration parameters before deciding where to place the VM and with which VM, but it consumes larger amount of energy. This is because not all jobs are exist based on the classification (or exist in small percentages). As a result, Many VMs exist with no corresponding VMs to be placed together on the same PM. In other words, the VMs of the jobs which intensively utilize specific type of resources possibly do not find the VMs of the jobs which intensively utilize the complement resources. The set of VMs with no complements to be place with them on the same PMs need extra computations to do the process of VM placement.

In the equal distribution subsets, the CI-DI approach also has a good performance compared with the other approaches. It is seen from Fig. 3 that the 16-8 approach outperforms the CI-DI approach. This is because the 16-8 approach has the best jobs distribution (exactly equal) among the subsets. Another important reason is that the approach combines the VMs allocated to jobs which are exactly utilize complement resources of the same PM. As a result, the involved PMs are utilized optimally, leading to a better energy efficiency.

IV. CONCLUSIONS

Cloud data centers consume large amounts of energy. One of the recent challenges in the cloud computing model is to enhance the energy efficiency in these data centers. This work tackles this issue by proposing and evaluating three energy aware job submission approaches. Each approach has its innovative job classification model, and its novel VM placement strategy which employs the jobs' subsets resulted from its corresponding classification model. The following notes can be concluded from the evaluation process:

1- In general, the CI-DI approach outperforms both 16-10 and 16-8 approaches due to the following reasons:
   - It always enforces the VMs of the jobs which requests different types of resources to be placed together on the same PM when possible. This maximizes the PMs utilization and, at the same time, minimizes the jobs influences on each other during their execution.
   - It gives the VMs of the CI jobs the choice to combine with the VMs of HDI and/or with the VMs of the BDI jobs. In both cases there is no conflict in the resources usages.
   - The subsets resulted from CI-DI approach is distributed in a good manner. The differences in the number of jobs between CI and DI jobs, and between CI-DI and Normal jobs are small compared with the differences in the numbers of jobs to be combined based on the 16-10 and 16-8 approaches. So, in CI-DI approach, VMs usually find their proper complements to be placed with on the same PM.

2- Placing the VMs allocated to jobs which utilize the same kind of resources negatively effects the energy efficiency.

3- Jobs that utilize the CPU and RAM resources are highly correlated.

4- The most effective parameter in energy consumption is the CPU.

As a future work, we are working to propose new methods to calculate other threshold values during jobs classification process, and studying the effects of the new thresholds, which will produce different subsets of jobs, on the resulted consumed energy. Moreover, the VM management concerns, such as VM migration and consolidation, will be applied to the proposed approaches to make them more integral to work in the cloud computing model.

REFERENCES

[1] S. Chaisiri, B. Lee and D. Niyato, "Optimal Virtual Machine Placement Across Multiple Cloud Providers," in *IEEE Asia-Pacific Services Computing Conference*, Singapore, 2009.

[2] H. Van, F. Tran and J. Menaud, "Performance and Power Management for Cloud Infrastructures," in *IEEE 3rd International Conference on Cloud Computing*, Miami, FL, 2010.

[3] F. Hermenier, X. Lorca, J. Menaud, G. Muller and J. Lawall, "Entropy: A Consolidation Manager for Clusters," in *ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments*, 2009.

[4] 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.

[5] F. Ma, F. Liu and Z. Liu, "Multi-objective Optimization for Initial Virtual Machine Placement in Cloud Data Center," *Journal of Information & Computational Science*, vol. 9, no. 16, pp. 5029-5038, 2012.

[6] A. Esnault, "Energy-Aware Distributed Ant Colony Based Virtual Machine Consolidation in IaaS Clouds," *HAL,* 2012.

[7] Y. Gao, H. Guan, Z. Qi, Y. Hou and L. Liu, "A Multi-Objective Ant Colony System Algorithm For Virtual Machine Placement in Cloud Computing," *Journal of Computer and System Sciences,* vol. 79, no. 8, pp. 1230-1242, 2013.

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

[9] M. H. Ferdaus, M. Murshed, R. N. Calheiros and R. Buyya, "Virtual Machine Consolidation in Cloud Data Centers Using ACO Metaheuristic," in *Euro-Par 2014 Parallel Proceedings of the 20th International Conference*, Porto, Portugal, 2014.

[10] D. Kumar and Z. Raza, "A PSO Based VM Resource Scheduling Model for Cloud Computing," in *IEEE International Conference on Computational Intelligence & Communication Technology*, Ghaziabad, 2015.

[11] S. Dashtia and A. Rahmania, "Dynamic VMs Placement For Energy Efficiency By PSO In Cloud Computing," *Journal of Experimental & Theoretical Artificial Intelligence,* 2015.

[12] N. Quang-Hung, P. Nienz, N. Namz, N. Tuong and N. Thoa, "A Genetic Algorithm for Power-Aware Virtual Machine Allocation in Private Cloud," *Information and Communication Technology,* vol. 7804, no. Lecture Notes in Computer Science, pp. 170-179, 2013.

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

[14] R. Calheiros, R. Ranjan, A. Beloglazov, C. 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,* vol. 41, no. 1, pp. 23-50, 2011.

[15] C. Reiss and J. Wilkes, Google Cluster-Usage Traces: Format and Schema, Google Inc., version 2, 2013.

[16] Amazon, 2016. [Online]. Available: http://aws.amazon.com.