Paving the Way for Energy Efficient Cloud Data Centers: A Type-Aware Virtual Machine Placement Strategy

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

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

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

***Abstract*: The rapid revolution of cloud computing model is accompanied by huge amounts of energy consumed by the cloud data centers. So, enhancing the energy efficiency of those data centers has become a major challenge. This paper tackles the problem of enhancing the energy consumption of cloud data centers by proposing a novel virtual machine placement strategy. The proposed strategy suits both static and dynamic placement process. It aims to better utilize the involved physical machines which host the virtual machines. 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 paper presents a mathematical formulation of the virtual machine placement process based on the Multiple Choice Knapsack Problem which is a generalization of the classical Knapsack Problem. The performance evaluation of the proposed strategy shows that it can enhance the energy efficiency of the cloud data centers by trying to minimize the number of the involved physical machines which host the virtual machines, and by optimally utilizing the involved physical machines.**

***Keywords*: Cloud Computing, VM Placement, Energy Efficiency.**

I. INTRODUCTION

Virtualization is the core concept of the cloud computing model. Virtualization is creating a virtual version of a real thing. It hides the details of the physical hardware layer from cloud users and provides virtualized resources for them. 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 a PM. It simulates the properties of a separated PM. The term *VM Management* refers to the process of coordinated provisioning of the virtualized resources. This process includes mapping virtual resources to physical resources in addition to overall management capabilities. Cloud providers are able to create multiple VM instances on a PM based on virtualization concept. Virtualization can improve the utilization of resources and thus reduces consumed energy. Many works tried to solve the problem of VM placement. The basic and simplest VM placement method is Round Robin, where VMs are placed on PMs unconditionally in a sequential manner. Another traditional approach to solve the problem of VM placement is linear programming, as in [1]. Different approach to solve this problem is by using constraint programming, as in [2] and [3]. Also, the bin packing (BP) approach is used in solving the problem of VM placement in cloud data centers. The thesis in [4] is an example of using BP to place VMs on PMs. In addition to the previous approaches, various optimization methods such as Ant Colony Optimization (ACO) [5] [6], Particle Swarm Optimization (PSO) [7] [8], and Genetic Algorithms (GA) [9] were used to solve the problems of VM placement and VM consolidation in the virtualized data centers.

This paper comes with the following contributions:

1) A novel VM placement strategy that considers the types of jobs which are served by the VMs before the process of VM placement. The strategy, which is Mixed Type Placement (MTP) and based on the Multiple Choice Knapsack Problem (MCKP) model, places VMs allocated to the jobs of different types on the same PM. It is used when the items have to be selected from different sets to be combined in one host. MTP is applicable for both initial VM placement, and for placing the migrated VM to fit in with the PM.

2) A detailed mathematical formulation of the VM placement process based on MCKP which is a generalization of the classical Knapsack Problem (KP).

The rest of this paper is organized as follows: Section II gives a detailed description about the VM placement process. Section III describes the definition of the KP and the definition of one of its generalization, which is the MCKP. It illustrates how the MCKP is modified to fit the proposed solution to the problem of VM placement process. The performance analysis of the proposed VM placement strategy is presented in section IV. Conclusions are listed in section V.

II. VIRTUAL MACHINE PLACEMENT

VM placement is the process of mapping the VMs into their best fit PMs. The problem of VM placement can be divided into two types; (1) ***Initial VM placement (Static)*** which is the process of placing a set of VMs at the same time on the PMs of the data center, considering requirements of VMs, capacities of PMs, and some other factors such as the consumed energy and the system performance, and (2) ***After Migration VM Placement (Dynamic)*** which is the process of re-placing

978-1-5090-5817-4/17 $31.00 © 2017 IEEE
DOI 10.1109/IC2E.2017.51

the VMs on other PMs to balance the system conditions, or to adapt with the changes in the VM requirements. This work proposes a novel model to solve the problem of VM placement. The model is based on the MCKP and it is suitable for both VM Placement types. MCKP is chosen because it is able to target two goals with one shot: (1) It tries to minimize the total number of the involved knapsacks by the process of items packing (in our case, every knapsack represents one PM) which minimize the consumed energy, and (2) It guarantees an acceptable utilization for the involved PMs, resulting in combining VMs which serve different types of jobs on the same PM whenever possible. Different types of jobs utilize different physical resources, and in this case, all resources are keeping busy, which in turn, enhance the PM utilization.

Minimizing the number of the involved PMs, which are utilized in an optimal way, leads to enhance the energy efficiency of the data center. In this work, PMs and VMs are represented as in the next sections:

A. *Physical And Virtual Machines Representation*

There is a set of m PMs in the cloud data center, $PM = \{pm_1, pm_2, \dots, pm_m\}$, and a set of n VMs which will be hosted on the PMs of the data center, $VM = \{vm_1, vm_2, \dots, vm_n\}$.

Concerning the PMs, each one has a limited capacity of the following resources: core(s), RAM, storage, and bandwidth, represented as:

$Cap_{core}(j)\ \forall j \in \{1,2,..m\}$ *represents the available cores of* pm_j
$Cap_{RAM}(j)\ \forall j \in \{1,2,..m\}$ *represents the available RAM of* pm_j
$Cap_{storage}(j)\ \forall j \in \{1,2,..m\}$ *represents the available storage of* pm_j
$Cap_{BW}(j)\ \forall j \in \{1,2,..m\}$ *represents the available bandwidth of* pm_j

So, the total capacity of the resources of any PM can be represented as:

$$PM_j^{Cap} = \left(Cap_{core}(j), Cap_{RAM}(j), Cap_{storage}(j), Cap_{BW}(j)\right)$$

And vector $(c_{ij}, r_{ij}, s_{ij}, b_{ij})$ represents the required number of cores, amount of RAM, size of storage, and size of bandwidth for a specific VM respectively. So, the total requirement of an instance VM_i from available infrastructure of PM_j can be represented as:

$$VM_i^{Req}(j) = \left(c_{ij}, r_{ij}, s_{ij}, b_{ij}\right)$$

B. *Virtual Machine Placement Representation*

According to the above PMs and VMs representations, the problem now is how to perform the VMs placement process for: *Initial VM placement*, *After Migration VM placement*. The problem of VM placement resulted from the mentioned two cases is represented and solved based on MCKP, which is a generalization of the classical KP, as discussed in details in the next sections. The VM placement process requires more than a single constraint (cores, RAM, storage, and bandwidth), so it is also multidimensional KP.

III. PROBLEM FORMULATION

There are different types of KP, all fall under the NP-hard class of problems [10]. The knapsack problem is the one that selects a subset of n items. The selection is done in such a way that the corresponding profit sum is maximized, while the weight sum does not exceed the capacity of knapsack. The problem of knapsack can be deployed to serve the problem of VM placement, and it is possible to use the fundamentals of knapsack problem to deal with VM placement. Each PM is considered as a knapsack, and each VM is considered as an item. The capacity of knapsack consists of the available cores, RAM, the storage, and the bandwidth of each PM. Each item has profit value and weight.

A. *The Standard Knapsack Problem*

The Knapsack problem can be described as: Given a set of items to be placed on a limited capacity Knapsack. Each item has a profit value and a weight. The problem is to choose specific items such that the sum is maximized/minimized for specific value, without having the weight sum to exceed the capacity of the knapsack. The standard KP can be formulated as [10]:

$$Max/Min \sum_{i=1}^{n} v_i\, x_i \qquad (1)$$

Subject to

$$\sum_{i=1}^{n} r_i\, x_i \le KnapsackCapacity \qquad (2)$$

$$i = 1,2,\dots,n$$

$$x_i \in \{0,1\}$$

Equations (1) and (2) represent the objective and constraint functions respectively. In KP, all coefficients $v, r, KnapsackCapacity$ are positive, v_i represents a value associated with each item (i.e. the values of the items to be maximized/minimized), r_i represents the required capacity by the item, and

$$x_i = \begin{cases} 1 & if\ the\ item\ placed\ on\ the\ Knapsak \\ 0 & otherwise \end{cases}$$

So, to solve the problem of VM placement, where VMs are mapped to their best PMs under the conditions of PMs limited capacity, the below equations are proposed to minimize the energy consumption:

$$Min \sum_{i=1}^{n} v_{i,j}\, x_{i,j} \qquad (3)$$

Subject to

$$\sum_{i=1}^{n} VM_i^{Req}\, x_{i,j} \le PM_j^{Cap} \qquad (4)$$

$$i = 1,2,\dots,n$$

$$x_{i,j} = \begin{cases} 1 & if\ VM_i\ placed\ on\ the\ PM_j \\ 0 & otherwise \end{cases}$$

$$\sum_{\forall\, i,j} x_{i,j} = 1$$

- $v_{i,j}$ is the amount of energy consumed when executing VM_i on PM_j which can be estimated as in the model presented in [11].

B. The Multiple Choice Knapsack Problem

The MCKP is a generalization of the classical KP. This generalization can be described as: Given T classes ($S_1, S_2, \dots, S_T$) of items to be placed on a limited capacity Knapsack. Each item i ∈ S_t has a profit value and a weight. The problem is to choose on item from each class such that the sum of a specific value is maximized/minimized, without having the weight sum to exceed the capacity of the knapsack. MCKP can be formulated as [10] [12]:

$$Max/Min \sum_{t=1}^{T}\sum_{i \in S_t} v_{ti}\, x_{ti} \quad (5)$$

Subject to

$$\sum_{t=1}^{T}\sum_{i \in S_t} r_{ti}\, x_{ti} \leq KnapsackCapacity \quad (6)$$

$$i = 1, \dots, n$$
$$i \in S_t$$
$$x_{ti} \in \{0,1\}$$

Equations (5) and (6) represent the objective and constraint functions respectively. In MCKP, all coefficients v, r, and $KnapsackCapacity$ are positive, their details are:

- v_{ti} is the value associated with each i_{th} item belong to the t_{th} class when it is placed on the knapsack (i.e. the values of the items to be maximized/minimized). As well, it can be estimated as in the model presented in [11].
- r_{ti} represents the required capacity by the item
- $x_{ti} = \begin{cases} 1 & if\ the\ item\ placed\ on\ the\ Knapsak \\ 0 & otherwise \end{cases}$
- Subsets or Classes ($S_1, S_2, \dots, S_T$) are mutually disjoint. Each class S_t has s_t items. The total number of items to be knapsacked (n) is equal to the sum the number of classes' items, $n = \sum_{t=1}^{T} s_t$.

Generally, in cloud data centers, VMs are owned by independent individuals or enterprises. This implies that the resulting workload on the cloud data centers is of mixed types of jobs. The mixed workload is formed by combining various types of applications, such as high performance computing applications, and web applications. These applications require and utilize the resources of the cloud data centers at the same time. High performance computing applications mostly utilize the compute resources (CPU and RAM), whereas web applications utilize the non-compute resources (storage and/or bandwidth). This results in better PMs utilization. The work in this paper employs the MCKP for the purpose of solving the problem of VMs placement. MCKP is deployed to be in line with the proposed MTP strategy. MTP combines the VMs allocated to different types of jobs on the same PM whenever possible. Upon their resources requirements, jobs can be divided into four subsets (classes) as described in [13]: *Compute Intensive* (CI) jobs which highly utilize compute resources, *Data Intensive* (DI) jobs which highly utilize storage and/or bandwidth resources, *Compute-Intensive Data-Intensive* (CIDI) jobs, which utilize compute resources together with storage and/or bandwidth resources all in a high manner, and *Normal* jobs, which not fit in any of the types specified above.

After classifying the jobs, a VM is allocated to each job in a best fit manner. Then, VMs allocated to jobs from type 1, type 2, type 3, and type 4 are gathered in four subsets S_1, S_2, S_3, S_4 respectively. As a next step, In the *initial VM placement*, the MCKP is applied to the subset ($S_1$and S_2), and to the subset (S_3 and S_4), to place their items (VMs) on the same knapsack (PM) wherever possible. The remaining VMs, which do not find their corresponding VMs based on MTP strategy, are combined in one set. Then, KP is applied to the VMs of this set to be placed on the Knapsacks. The involved PMs resulted from the VM placement process are used optimally, because they host VMs which request different kinds of resources. In such case, the PMs resources are kept busy and consequently the PMs utilization is maximized. While in *the after migration VM placement*, MCKP is applied to the subset ($S_1$and S_2), and to the subset (S_3 and S_4), to place their items (VMs) on the same knapsack (PM) wherever possible. If the VMs do not find their corresponding VMs based on MTP strategy, the classical KP is applied to the items of this set to be placed on the Knapsack. This process is exactly similar to MCKP. In MCKP, the set of items is classified and then, instead of taking two or more items from the same set, one item is selected from each class to be packed in the knapsack [12]. The MCKP can be rewritten as below to be in line with the proposed VM placement strategy:

$$Min \sum_{t=1}^{T}\sum_{i \in S_t} v_{ti,j}\, x_{ti,j} \quad (7)$$

Subject to

$$\sum_{t=1}^{T}\sum_{i \in S_t} VM_{ti}^{Req}\, x_{ti,j} \leq PM_j^{Cap} \quad (8)$$

$$i = 1, \dots, n$$
$$i \in S_t$$
$$x_{ti,j} = \begin{cases} 1 & if\ VM_i \in S_t\ placed\ on\ the\ PM_j \\ 0 & otherwise \end{cases}$$
$$\sum_{\forall\, ti,j} x_{ti,j} = 1$$

- $v_{ti,j}$ is the value of the i_{th} item of the t_{th} class when it is knapsaked on PM_j. In our case, it is the consumed energy when place VM_i ∈ Class *t* on PM_j .

Each item from the four subsets has a particular value and it requires some resources. The objective of the MCKP is to pick exactly one (only one) item (VM) from each subset to minimize the total value of the combined items, subject to the resource constraints (capacity) of the knapsack (PM).

The indexes in the proposed formulation can be separated into two parts: Part related to VMs (*represented by variables t and i*), and Part related to PMs (*represented by variable j*).

The part that related to VMs is separated into two sub-parts: Sub-part related to the jobs types (*represented by variable t*), and Sub-part related to the index of the VM in the jobs types subset (*represented by variable i*)

So, variable t is responsible for ensuring placing VMs allocated to of different job types on the same PM.

IV. Performance Analysis

The experiment in this section is done to show the effects of considering the jobs types during the VM placement process on the consumed energy. The experiments are repeated for a different number of VMs allocated to a submitted set of jobs, from 200 to 1000 jobs. Both the classical KP and the MCKP are applied to solve the problem of the initial VM placement. Figure 1 shows that applying the MCKP (which consider the jobs types) is more energy-efficient than the classical KP (which do not consider the jobs types) in the VM placement.

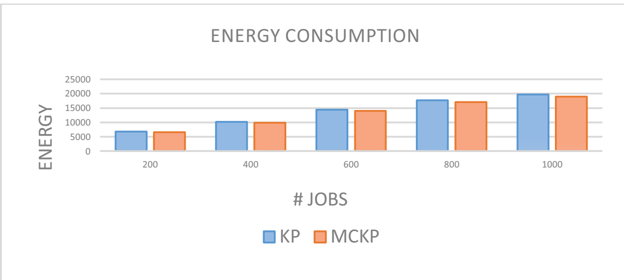

Fig. 1: The consumed energy when applying KP and MCKP in the VM placement process.

In the *Initial VM placement*, the proposed model enhances the energy efficiency by trying to minimize the total number of the involved PMs which served the users' jobs, and maximized the utilization of the involved PMs. The minimum possible number of the PMs is guaranteed by KP optimization solver model. KP places the maximum number of the possible packed items (VMs) on every knapsack (PM). The better utilization for the resulted involved PMs comes up because MCKP model places the VMs allocated to different types of jobs on the same PM whenever possible. In this case, all the PM resources are keeping busy because the VMs allocated to different types of jobs are placed on the same PM, and every job utilizes different kind of resources. This maximizes the PM utilization, and consequently enhances energy efficiency. Figure 2 shows the PMs utilization when applying MCKP and KP.

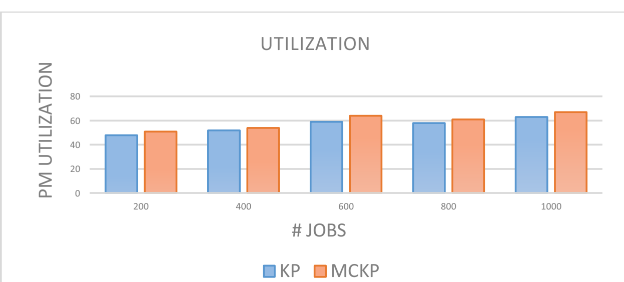

Fig. 2: The involved PMs utilization when applying KP and MCKP in the VM placement process.

In the *after migration VM placement*, as in the initial placement, MCKP model places the migrated VM on a PM that hosts a VM allocated to different kinds of jobs whenever possible. Thus, the resources of the PM which are not utilized well will be utilized better by keeping them busy during the PM involved period.

V. Conclusions

This paper tackles the problem of high energy consumption of cloud data centers by proposing a novel VM placement strategy, called MTP. MTP employs MCKP to better utilize the involved PMs which host the VMs. From the energy efficiency perspective, employing MCKP in the VM placement process outperforms employing KP. This is due to the better optimal use of the resources of the involved PMs.

References

[1] B. Speitkamp and M. Bichler, "A mathematical programming approach for server consolidation problems in virtualized data centers," *IEEE Transactions on Services Computing,* vol. 3, no. 4, pp. 266-278, 2010.

[2] H. Van, F. Tran and J. Menaud, "Performance and power management for cloud infrastructures," in *IEEE 3rd International Conference on Cloud Computing*, 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 & Information Systems, The University of Melbourne, 2013.

[5] 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*, 2014.

[6] A. Ashraf and I. Porres, "Multi-objective dynamic virtual machine consolidation in the cloud using ant colony system," *International Journal of Parallel, Emergent and Distributed Systems,* no. arXiv:1701.00383 [cs.DC], 2017.

[7] D. Kumar and Z. Raza, "A PSO based VM resource scheduling model for cloud computing," in *IEEE International Conference on Computational Intelligence & Communication Technology*, 2015.

[8] S. Dashtia and A. Rahmania, "Dynamic VMs placement for energy efficiency by PSO in cloud computing," *Journal of Experimental & Theoretical Artificial Intelligence,* 2015.

[9] 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.

[10] D. Pisinger, Algorithms for knapsack problems, PhD thesis, University of Copenhagen, 1995.

[11] X. Fan, W. Weber and L. Barroso, "Power provisioning for a warehouse-sized computer," in *the 34th Annual International Symposium on Computer Architecture*, 2007.

[12] H. Kellerer, U. Pferschy and D. Pisinger, "The multiple-choice knapsack problem," in *Knapsack Problems*, Springer, pp. 317-347, 2004.

[13] A. Al-Dulaimy, R. Zantout, W. Itani and A. Zekri, "Job submission in the cloud: energy aware approaches," in *The World Congress on Engineering and Computer Science*, 2016.