Energy-Efficient Virtual Machine Replication and Placement in a Cloud Computing System

Hadi Goudarzi and Massoud Pedram
University of Southern California
Department of Electrical Engineering
{*hgoudarz, pedram*}@usc.edu

ABSTRACT – By utilizing Virtual Machines (VM) and doing server consolidation in a datacenter, a cloud provider can reduce the total energy consumption for servicing his clients with little performance degradation. In particular, the cloud provider can take advantage of dissimilar workloads and by assigning these workloads to the same server, can utilize fewer active servers to service his clients. Placing multiple copies of a VM on different servers and distributing the incoming requests among these VM copies can reduce the resource requirement for each VM copy and help the cloud provider utilize the servers more efficiently. In this paper, the problem of energy-efficient VM placement in a cloud computing system is solved. Precisely, we present an approach that first creates multiple copies of VMs and then uses dynamic programming and local search to place these copies on the physical servers. Simulation results show that the proposed algorithm reduces the total energy consumption by up to 20% with respect to previous work.[1]

I. INTRODUCTION

Demand for computing power has been increasing due to the penetration of information technologies in our daily interactions with the world both at personal and public levels, encompassing business, commerce, education, manufacturing, and communication services. At the personal level, the wide scale presence of online banking, e-commerce, SaaS (Software as a Service), social networking and so on produce workloads of great diversity and enormous scale. At the same time computing and information processing requirements of various public organizations and private corporations have also been increasing rapidly. Examples include digital services and functions required by various industrial sectors, ranging from manufacturing to housing, from transportation to banking. Such a dramatic increase in the computing demand requires a scalable and dependable information technology (IT) infrastructure comprising of servers, storage, network bandwidth, physical infrastructure, Electrical Grid, IT personnel and billions of dollars in capital expenditure and operational cost to name a few.

Virtualization technology makes the independence of applications and servers feasible. Nowadays, computing systems heavily rely on this technology. Virtualization technology provides a new way to improve the power efficiency of the datacenters: (server) consolidation, which enables the assignment of multiple virtual machines (VMs) to a single physical server. By this action, some of the available servers can be turned off or put into some deep sleep state, thereby, lowering power consumption of the computing system. The technique works because modern servers tend to consume 50% or so of their peak power in idle state (this effect is known as the non-energy-proportionality of modern servers [1].) Consolidation involves performance-power tradeoff. More precisely, if workloads are consolidated on servers, performance of the consolidated VMs may decrease because of reduction of available physical resources (CPU, memory, I/O bandwidth) although the overall power efficiency improves because fewer servers are needed to service the VMs.

Low utilization of servers in a datacenter is one of the most important factors responsible for low power efficiency of datacenters. For example, the average utilization of servers in a Google datacenter was reported to be 30% [2]. Due to the non-energy-proportional nature of the current servers, it is prudent from an energy efficiency viewpoint to have as few servers as possible turned on, with each ON server being highly utilized. Hence, there is a strong justification for server consolidation in current enterprise computing centers.

The IT infrastructure provided by the datacenter owners/operators must meet various Service Level Agreements (SLAs) established with the clients. The SLAs may be resource related (e.g., amount of computing power, memory/storage space, network bandwidth), performance related (e.g., service time or throughput), or even quality of service related (24-7 availability, data security, percentage of dropped requests.) To minimize the energy consumption using consolidation, these SLA constraints should be considered.

A datacenter comprises of thousands to tens of thousands of server machines, working in tandem to provide services to the clients, see for example reference [1], [3] and [4]. In such a large computing system, in spite of non-energy-proportional characteristics of current server machines, energy efficiency can be maximized through system-wide resource allocation and server consolidation. The problem of resource provisioning is challenging due to the diversity that is present in the hosted client applications. For example: some client applications may be computation-intensive while others may be memory-intensive, some applications may run well together while others do not, and so on.

[1] This work is supported in part by a grant from the National Science Foundation, CISE Directorate.

The energy cost and admission control policy in a cloud computing system are affected by its power and VM management policies. Power management techniques [3]-[7] control the average and peak power in a distributed or centralized fashion in the datacenters VM management techniques [8]-[11] control the VM placement on the physical servers as well as VM migration from one server to next. In this paper, we focus on the VM placement to minimize the energy cost in the cloud computing system.

Generating multiple copies of a VM and placing them on different servers is one of the basic ways to increase the service reliability. In this approach, only the original copy of the VM handles the requests while the other copies are idle. In this paper, we propose to exploit all of these VM copies for servicing the requests. Under this new scenario, resource provided to each copy of the VM should satisfy SLA requirements and the set of distributed VMs should be able to service all of the incoming requests. For this reason, memory Bandwidth (BW) provided for each copy of the VM should be the same as the original VM and the total CPU cycles provided for all of the VM copies should be equal to those provided to the original copy of the VM. Increasing the number of VM copies increases the average utilization level of servers because by increasing the number of VM copies, we have more opportunity to use smaller VMs to fully utilize the servers, and thereby, avoid having under-utilized servers. Using this approach and an effective VM placement algorithm, which determines the number of VMs and places them on the physical machines, the energy cost of the system can be reduced by up to 20% (cf. Section VI.)

The proposed VM placement algorithm is based on the dynamic programming and local search methods. The dynamic programming method determines the number of copies for each VM and places them on servers whereas the local search tries to minimize the energy cost by turning off the under-utilized servers.

The remainder of this paper is organized as follows. Related work is presented in the next section. The system model and problem formulation are given in section III and IV. The proposed algorithm is presented in section V. Simulation results are provided in the section VI and the conclusions are stated in the last section.

II. Related work

Distributed resource allocation is one of the most challenging problems in the resource management field. This problem has attracted a lot of attention from the research community in the last few years. In the following we provide a review of most relevant prior work.

Srikantaiah et al. [12] presented an energy-aware consolidation technique to decrease the total energy consumption of a cloud computing system. The authors empirically modeled the energy consumption of servers as a function of CPU and disk utilization. Next, they described a simple heuristic to consolidate the processing works in the cloud computing system. Performance of the solution is evaluated only for very small input size.

A VM placement heuristic to maximize the number of serviced applications, minimize the migration cost, and balance the load in physical machines is presented in [8]. The main focus of this work is on the scalability of the problem but the problem of assigning VMs on physical servers when the operational cost minimization is the objective is not investigated.

A consolidation manager for homogenous clusters called Entropy is proposed in reference [9]. This consolidation manager considers the memory and CPU requirements of the VMs and migration overhead in the system. The status of each VM (being active or inactive) can change in time and this signifies the importance of having dynamic consolidation manager. The authors proposed a dynamic consolidation based on constraint programming methods considering the performance overhead of migrating (active and inactive) VMs to different servers.

Bobroff et al. [10] proposed methods for classification and forecasting of VM workload. It is shown that applying dynamic VM migration is most beneficiary in case of certain workload characteristics. Based on these methods, they proposed a dynamic VM management algorithm to minimize the total power consumption with a constraint on SLA of each VM or minimize the SLA violation rates considering a fixed set of active servers.

Power and migration cost aware application placement in virtualized systems is proposed in [11]. Authors present a power-aware VM placement controller in a system with heterogeneous server clusters and virtual machines. pMapper architecture and placement algorithms to solve the problem of minimizing power subject to a fixed performance requirement are investigated. The proposed solution is presented based on assumption of predetermined performance level for VMs. The proposed algorithms for VM placement, which are based on bin packing heuristics, do not consider multiple copies of VM and only consider one dimension of resource in the servers.

Liu et al. [13] described a SLA-based profit optimization problem in electronic commerce hosting datacenters. A fixed set of servers are assumed to be active and application placement on the servers are done to maximize the total SLA profit. SLA in this work is modeled as a response time constraint and less than a portion (e.g. 2%) of request's response time can violate that constraint. This kind of SLA are used in [14] to optimize the energy consumption and migration cost in the cloud computing system.

Zhang et al. [15] and Ardagna et al. [16] present heuristics to allocate resources in a virtualized environment to maximize the profit and minimize the energy cost in the system while meeting the SLA. The authors used a complex model for energy calculation to increase the accuracy. They solved the problem by generating a feasible solution and improving the quality of the solution by local search. The presented problem considers soft SLA contracts in which client pays the cloud provider based on the average response time provided to its requests. These kind of SLAs are considered in different works such as [17]-[19]. The VM placement problem with constant resource requirements for each VM is not considered in these works.

Our paper considers the resource management problem in a cloud computing system. Key features of our formulation and proposed solution are that we consider heterogeneous servers in the system and use a two dimensional model of the resource usage accounting for both computational and memory BW. We propose multiple copies of VMs to be active in each time to reduce the resource requirement for each copy of VM and help to increase the energy efficiency of the consolidation and VM placement algorithm. A novel approach based on dynamic programming and local search is proposed to determine the number of copies for each VM and place them on servers to minimize the total cost in the system. No previous work considers all these aspects together when addressing the cloud level resource management problem.

III. System Model

In this section, detail of the assumptions and system configuration for the VM placement problem are presented.

To increase the paper readability, Table I presents key symbols and definitions used in this paper. Each client is identified by a unique id, denoted by index *i*. Each server in the cloud computing system is similarly identified by a unique id, denoted by index *j*.

Table I. Notation and Definitions

Symbol name	*Definition*
c_i^m, c_i^p	Required memory BW and total processing capacity for the i[th] client
L_i	Max. # of servers allowed to serve the i[th] client
s_k	Set of servers of type k
C_j^p, C_j^m	Total CPU cycle and memory BW of the j[th] server, shorthand notation for $C_{S_k}^p$ and $C_{S_k}^m$
P_j^0	Constant power consumption of the j[th] server operation in the active mode. Same as $P_{S_k}^0$
P_j^{p}	Power of operating the j[th] server which is proportional to the utilization of processing resources, shorthand notation for $P_{S_k}^p$
T_e	Duration of a *decision epoch* in seconds
x_j	A pseudo-Boolean integer to determine if the j[th] server is ON (1) or OFF (0)
y_{ij}	A pseudo-Boolean integer to determine if the i[th] VM is assigned to the j[th] server (1) or not (0)
ϕ_{ij}^p, ϕ_{ij}^m	Portion of the processing and memory BW resources of the j[th] server that is allocated to the i[th] client
ϕ_j^p, ϕ_j^m	Portion of the processing and memory BW resources of the j[th] server that is allocated to any client

A. Cloud Computing System

In the following paragraphs, we describe the type of the datacenter that we have assumed as well as our observations and key assumptions about where the performance bottlenecks are in the system and how we can account for the energy cost associated with a client's VM running in the datacenter.

A datacenter comprises of a number of potentially heterogeneous servers chosen from a set of known and well-characterized server types. In particular, servers of a given type are modeled by their processing capacity or CPU cycles (C_*^p) and memory BW (C_*^m) as well as their energy cost, which is directly related to their average power consumption. We assume that local (or networked) secondary storage (disc) is not a system bottleneck.

The operational cost of the system is assumed to be the total energy cost of serving all clients' requests. The energy cost is calculated as the server power consumption multiplied by the duration of the epoch in seconds (T_e). The power of a server is modeled as a constant power cost (P_*^0) plus another variable power cost, which is linearly related to the utilization of the server (with slope of P_*^p). Note that power cost of communication resources and the computer room air conditioning (CRAC) units are amortized over all servers and communication/networking gear in the datacenter, and are thus assumed to be relatively independent of the clients' workloads. Precisely, these costs are not included in the equation for datacenter power cost.

B. Client and Virtual Machines

Clients in the cloud computing system are represented with VM. By using workload prediction with consideration of the SLA, the amount of resources that each client needs in order to avoid SLA violation can be determined. These VMs are assumed to generate processing requests during the considered epoch. This assumption is valid for clients corresponding to online services. In contrast, this assumption is not applicable for batch applications.

Each VM may be copied onto different servers (i.e., requests generated by a single client can be assigned to more than one server). This request distribution can decrease the quality of the service if the number of servers that process the client's requests becomes large [15]. Therefore, in this paper, we impose an upper bound on this number; L_i limits the maximum number of copies of the VM in datacenter (this limit is set to 1 if the VM cannot have multiple copies). If multiple copies of a VM are placed on different servers, the following constraints should be satisfied:

$$\sum_j \phi_{ij}^p C_j^p = c_i^p \quad (1)$$

$$\phi_{ij}^m y_{ij} C_j^m = c_i^m \quad (2)$$

where ϕ_{ij}^p and ϕ_{ij}^m denote the portion of the j[th] server CPU cycles and memory BW allocated to the VM related to the i^{th} client. Moreover, y_{ij} is a pseudo-Boolean integer to determine if a VM related to client *i* is assigned to the j^{th} server or not. Constraint (1) enforces the summation of the reserved CPU cycles on the assigned servers to be equal to the required CPU cycles for client *i*. Constraint (2) enforces the provided memory BW on assigned servers to be equal to the required memory BW for the original VM. This constraint enforces the cloud provider not to sacrifice the Quality of Service (QoS) for its clients. An example of multiple copies of a VM is shown in Figure 1. In this figure, the difference between heights of the horizontal bars shows the memory BW requirement while the difference between widths of the vertical lines show the CPU cycle requirement of each VM.

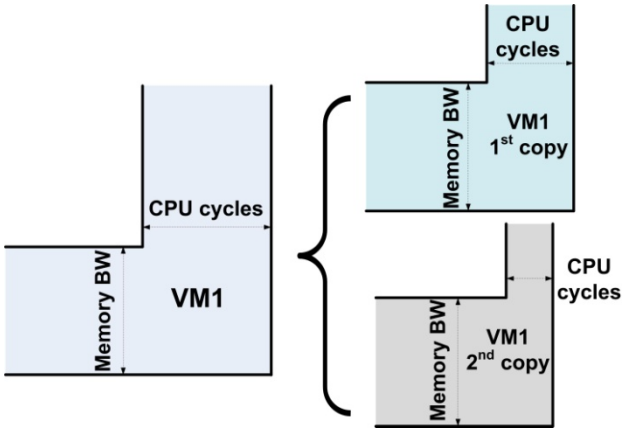

Figure 1. An example of multiple copies of a VM

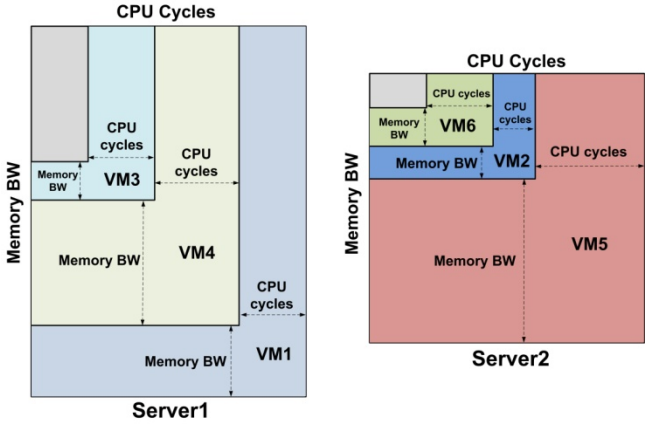

Figure 2. An exemplary solution for assigning six heterogeneous VMs on two heterogeneous Servers

C. VM Management System

Datacenter management is responsible for admitting the VMs into the datacenter, servicing them to satisfy SLA requirements, and minimizing the energy cost of the datacenter. In this paper, we focus on the *VM controller* (VMC). The VMC is responsible for determining the resource requirements of the VMs and placing them on servers. Moreover, to mimic the workload changes, the VMC should perform VM migration. The VMC performs these tasks based on two different optimization procedures: semi-static optimization and dynamic optimization. Semi-static optimization procedure is performed periodically (at periods of T_e), whereas dynamic optimization procedure is performed whenever it is needed.

In the semi-static optimization procedure, the VMC considers the active set of VMs, previous assignment solution, feedbacks generated from power, thermal and performance sensors, and workload prediction to generate the best VM placement solution for the next epoch. Period of performing semi-static optimization depends on the type and size of the datacenter and workload specifications. In the dynamic optimization procedure, the VMC finds a temporary VM placement solution by migrating, creating or removing some number of VMs to respond to performance, power budget, or critical temperature violation.

In this paper, we focus on the semi-static optimization procedure of the VMC. In this procedure, resource requirements of VMs are assumed to be determined based on the SLA specification and workload estimation for the next decision epoch. The duration of the decision epoch is long enough for us to neglect the task migration penalty (less than 100ms for live migration [11]) with respect to the gain of the global optimization. Therefore, the energy cost optimization is performed without any knowledge of the state of the cloud computing system in the previous decision epoch. However, the nature of the proposed solution allows the system to account for the migration cost if this becomes an important consideration.

The role of the semi-static optimization procedure in the VMC is to determine whether to create multiple copies of VMs on different servers and assign VMs to servers. Considering fixed payments by the clients for the cloud services they receive, the goal of this optimization is to minimize the energy cost of the active servers in datacenter. An exemplary solution for assigning six VMs with different resource requirements on two heterogeneous servers is depicted in Figure 2.

IV. Problem Formulation

In this paper, a VM placement problem is considered with the objective of minimizing the total energy consumption in a decision epoch while servicing all VMs in the cloud computing system.

The exact formulation of the aforesaid problem (called MERA for Multi-dimensional Energy-efficient Resource Allocation) is provided below (cf. Table I.)

$$Min \quad T_e \sum_j x_j \left(P_j^0 + P_j^p \sum_i \phi_{ij}^p \right) \tag{3}$$

subject to:

$$\phi_j^p = \sum_i \phi_{ij}^p \leq 1 \qquad \forall j \tag{4}$$

$$\phi_j^m = \sum_i \phi_{ij}^m \leq 1 \qquad \forall j \tag{5}$$

$$\sum_j C_j^p \phi_{ij}^p = c_i^p \qquad \forall i,j \tag{6}$$

$$y_{ij} \geq \phi_{ij}^p, \quad y_{ij} \leq 1 + \phi_{ij}^p - \varepsilon \qquad \forall i,j \tag{7}$$

$$\phi_{ij}^m y_{ij} C_j^m = c_i^m \qquad \forall i,j \tag{8}$$

$$\sum_i y_{ij} \leq L_i \qquad \forall i \tag{9}$$

$$x_j \geq \sum_i \phi_{ij}^p \qquad \forall j \tag{10}$$

$$y_{ij} \in \{0,1\}, x_j \in \{0,1\}, \phi_{ij}^p \geq 0, \phi_{ij}^m \geq 0 \qquad \forall i,j \tag{11}$$

where ε is a very small positive value, and, x_j is a pseudo-Boolean integer variable to determine if the j[th] server is ON (x_j=1) or OFF (x_j=0).

The objective function is the summation of the energy cost of the ON servers based on a fixed power factor and a variable power term linearly related to the server utilization. In this problem, x_j, y_{ij} and ϕ_{ij}^p are the optimization variables.

The constraints capture the limits on the number of available servers and clients. In particular, inequality constraints (4) and (5) represent the limit on the utilization of the processing and memory BW in the j[th] server, respectively. Constraint (6) ensures that required processing capacity for each VM is provided. Constraint (7) generates a pseudo-

Boolean parameter that determines if a copy of a VM is assigned to a server ($y_{ij} = 1$) or not ($y_{ij} = 0$). Constraint (8) ensures the memory BW needs of a VM assigned to a server is provided and constraint (9) ensures that the number of copies of a VM does not exceed the maximum possible number of copies. Constraint (10) generates the pseudo-Boolean parameter related to the status of each server. Constraint (11) specifies the domains of optimization variables.

Theorem I: Generalized Assignment Problem (GAP) [20] can be reduced to the MERA problem.

Proof: Consider a version of the MERA problem in which $cost_j^0$ is equal to zero for every server and L_i is equal to one for every VM. In this problem, assigning each VM (only one copy) to each server has different cost and each server has two dimensional resources that can be assigned to VMs. So, we can solve any two-dimensional GAP problem using MERA problem's solution in a special case. ■

Theorem II: Bin Packing Problem (BPP) [20] can be reduced to MERA problem.

Proof: Consider a version of the MERA problem in which $cost_j^p$ is equal to zero for every server and L_i is equal to one for every VM. The objective in this problem is to minimize the number (or a weighted summation) of the ON servers considering the two-dimensional resource availabilities on each server and resource requirement of each VM. So, we can solve any two-dimensional BPP problem using MERA problem's solution in a special case. ■

Theorem I and II shows that MERA is a combination of two NP-hard problems. Considering either theorem I or II, MERA is at least as hard as a known NP-hard problem [20] and thus it is an NP-hard problem. Indeed, even deciding whether a feasible solution exists for this problem, does not have an efficient solution. In this paper, we consider a case in which the required resources for VMs are smaller than the available resources in the datacenter. This means we consider energy minimization with a fixed set of VMs instead of maximizing the number (or the total profit) of VMs that are served in the datacenter. So, we assume that a simple greedy algorithm (similar to First Fit Decreasing (FFD) heuristic [20]) will find a feasible solution to MERA for the specified inputs in the problem definitions. Another important observation about this problem is that the numbers of clients and servers in this problem are very large; therefore, a critical requirement for any proposed heuristic should be its scalability.

Different versions of this problem are considered in the literature. Some work consider this resource allocation problem along with performance modeling by using the queuing theory while other work considers a fixed size VM placement problem. In most of the previous work, simple solutions based on known heuristics for well-known problems such as Knapsack problem, bin packing, and generalized assignment problem are proposed. In this work we examine the effect of multiple active copies of VMs and an effective algorithm to reduce the energy consumption in the cloud computing system is proposed.

V. Proposed Algorithm

In this section, a heuristic for solving the MERA problem is presented. An algorithm based on Dynamic Programming (DP) is presented to determine the number of copies of each VM and assign these VMs to the servers. This decision determines which servers are active for the next epoch and the utilization of those servers. The goal of the algorithm is to minimize the total energy cost of the active servers. To improve the results, a local search is considered to minimize the number of active servers as much as possible.

In the beginning of the VM placement, clients are ordered based on their processing requirement. Based on this ordering, the optimal numbers of copies of the VMs are determined and these copies are placed on servers using dynamic programming. In the local search method, servers are turned off based on their utilization and VMs are placed on the rest of the servers (if possible) to minimize the energy consumption as much as possible.

The details of the Energy-efficient VM Replication and Placement algorithm or EVRP for short are presented below.

A. Energy Efficient VM Placement Algoritmh

At the beginning of the algorithm, ϕ_j^p and ϕ_j^m for each server are set to zero. A constructive approach is used to place VMs on the servers. VMs are sorted based on their processing requirements in a non-increasing order. For each VM, a method based on DP is used to determine the number of copies placed on different servers.

Energy cost of assigning a copy of the i[th] VM to a server from server type k is calculated based on equation (12).

$$c_{ik}(\alpha) = \phi_{ij}^p P_j^p + P_j^0 c_i^m / C_j^m \quad (12)$$

where α (between 1 and L_i) denotes the size of the assigned VM to the server. ϕ_{ij}^p is calculated from equation (13).

$$\phi_{ij}^p = (\alpha c_i^p / L_i) / C_j^p \quad (13)$$

For example, in case of L_i=4 if half of the CPU cycle requirement of the VM is provided by a copy of the VM, α and ϕ_{ij}^p is equal to 2 and $0.5c_i^p/C_j^p$.

The first term in (12) is the cost related to the CPU utilization of the server. The second term is the replacement of the constant energy cost of the active server.

For each VM, equation (12) is calculated for each server type and different values of α (between 1 and L_i). Moreover, for each server type, L_i active servers and L_i inactive servers that can service at least the smallest copy of the VM are selected as candidate hosts. For active servers, the value of cost is decremented by ε to select them over inactive servers in an equal energy scenario.

After selecting active and inactive candidate servers for each server type and calculating cost for each possible assignment, the problem is reduced to (14).

$$Min \ \sum_{j \in P} y_{ij}^{\alpha} c_{ij}(\alpha) \quad (14)$$

subject to:

$$\sum_{j \in P} \alpha y_{ij}^{\alpha} = L_i \quad (15)$$

where y_{ij}^{α} denotes the assignment parameter for j[th] server with VM with size of α (1 if assigned and 0 otherwise). Moreover, P denotes the set of candidate servers for this assignment.

DP is used to solve this problem and find the best assignment decision. In this DP method, candidate servers may be processed in any order. The method examines all the possible VM placement solution efficiently without calculating every possible solution in a brute-force manner. Using this method, the optimal solution can be found.

Complexity of this DP method is $O(2L_i^2K)$, where K denotes the number of server types that are considered for this assignment. After finding the assignment solution, ϕ_j^p and ϕ_j^m of the selected servers are updated. Then, the next VM is chosen and this procedure is repeated until all VMs are placed.

Algorithm 1 shows the pseudo code for this assignment solution for each VM.

Algorithm 1: Energy Efficient VM Placement

Inputs: C_j^m, C_j^p, P_j^0, P_j^p, c_i^m, c_i^p, L_i

Outputs: ϕ_{ij}^p, ϕ_{ij}^m (i is constant in this algorithm)

```
P={}
For (k = 1 to number of server types)
    ON=0; OFF=0;
    For (α = 1 to L_i)
        φ_ij^p = (α c_i^p / L_i) / C_j^p
        c_ik(α) = φ_ij^p P_j^p + P_j^0 c_i^m / C_j^m
    End
    J^ON = {j ∈ s_k | (1 − φ_j^m) ≥ c_i^m / C_j^m}
    J^OFF = {j ∈ s_k | φ_j^p = 0, (1 − φ_j^m) ≥ c_i^m / C_j^m}
    Foreach (j ∈ s_k)
        If (j ∈ J^ON & ON< L_i)
            P = P ∪ {j} , ON++, cost_ij(α) = c_ik(α) − ε
        Else if (j ∈ J^OFF & OFF< L_i)
            P = P ∪ {j} , OFF++, cost_ij(α) = c_ik(α)
    End
End
X = L_i, and Y = size (P)
Foreach ( j ∈ P)
    For (x = 1 to X)
        D[x,y]= infinity; //Auxiliary X×Y matrix used for DP
        For (z = 1 to x)
            D[x,y]=min(D[x,y],D[x-1,y-z]+ cost_ij(z))
        D[x,y]=min(D[x,y], D[x-1,y])
    End
End
Back-track to find the best assignment and update φ_j's
```

B. Local Search Method

The constructive nature of the proposed algorithm can cause a situation in which some servers are not well utilized although with the large number of VMs makes this less of a concern. Regardless, to improve the results of the proposed VM placement algorithm, a local search method is used.

To minimize the total energy consumption in the system, all servers with utilization less than a threshold are examined. This threshold can be specified by the cloud provider. To find the under-utilized servers, utilization of a server is defined as the maximum resource utilization in different resource dimensions in the server, e.g. if ϕ_j^p =0.5 and ϕ_j^m =0.3, we define the utilization of the server to be 50%.

To examine these under-utilized servers, each of them is turned off one by one and total energy consumption is found by placing their VMs on other active servers using the proposed DP placement method. If the total cost of the new placement is less than the previous total cost, the new configuration is selected and the remaining under-utilized servers are examined; otherwise the option of turning off that server is rejected and other candidate servers are examined.

VI. Simulation Results

To evaluate the effectiveness of the proposed VM placement algorithm, a simulation framework is implemented. Simulation setups, baseline heuristics and numerical results of this implementation are presented in this section.

A. Simulation Setup

For simulation, model parameters are generated based on the real world parameters. The number of server types is set to 8. For each server type, an arbitrary number of servers are placed in datacenter. Processors in server types are selected from a set of Intel processors (e.g. Atom and Xeon) [21] with different number of cores, cache, power consumptions and working frequencies. Active power consumptions for different server types (excluding the processor power consumption) are set uniformly between two to four times the power consumption of their fully-utilized processor. Memory BW of the servers is set based on the maximum memory BW of these cores with a factor of 0.4. For example if the maximum memory BW of a processor is 20 GB/s, the available memory BW for this processor is set to 8 GB/s.

Processing resource requirement for each VM is selected uniformly between 1 and 18 billion CPU cycles per second. The memory BW requirements for clients are also selected uniformly between 768MB/s and 4GB/s. Note that we limited the CPU cycle requirement of each VM to the maximum available CPU cycles in servers to be able to compare our proposed approach with the previous VM placement solution that are not able to create multiple VM copies automatically. In contrast, EVRP algorithm is capable of generating a VM placement solution as long as the memory BW requirement of each VM is less than the maximum memory BW supported by the available servers in the datacenter.

Upper bound on the number of copies for each VM is set between 1 and 5 based on the value of the required processing resources, e.g. if the processing requirement for a VM is equal to maximum processing requirements, L_i is equal to 5 and if the value of processing requirement for a VM is less than ¼ of the maximum value, L_i is equal to one (no copy is allowed).

Each simulation is repeated at least 1000 times to generate acceptable average results for each case.

B. Heuristics for Comparison

We implemented the min Power Parity (mPP) heuristic [11] as one of the state of the art energy-aware VM placement techniques. This heuristic is based on first fit decreasing heuristic [20] for the bin-packing problem. The heuristic tries to minimize the overall power consumed by the active servers in datacenter servicing the VMs. Details of mPP can be found in [11].

To show the effectiveness of our proposed approach for placing multiple copies of VMs on servers, along with mPP, a version of our algorithm in which every L_i is set to one is also considered. This prohibits the solution from using more than a

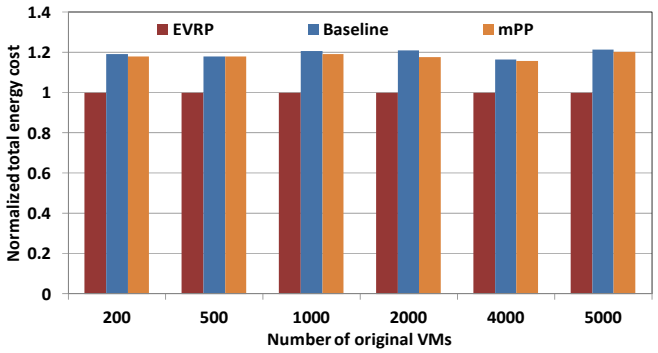

Figure 3. Normalized total energy cost of the system

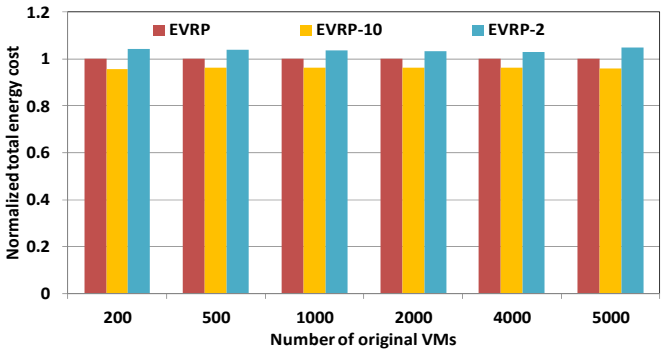

Figure 4. Normalized total energy cost of the VM placement solution using for different L_i

single VM per client. We denote this version of the algorithm with the name of baseline method in the figures. Moreover, to show the effect of distributed resource assignment and constant power cost for active servers, we implement a procedure to find the lower bound on the total energy cost with relaxation of these obstacles. To calculate this lower bound, for each VM, total energy cost $(c_i^p/C_j^p(P_j^p + P_j^0)T_e)$ of serving that VM on each server is calculated and the smallest energy cost is selected. Summation of these energy costs generates a lower bound on the total energy cost.

C. *Numerical Results*

Normalized total energy cost in the system using the EVRP algorithm, baseline method, and mPP algorithm is presented in Figure 3.

As can be seen, EVRP reduces the total energy cost of VM placement solution by 16 to 20% with respect to mPP algorithm. Performance of the baseline algorithm which is based on assigning the VMs using DP method is 1 to 4% worse than mPP method because baseline method does not place the VM on the server with least resource availability and instead choose the host server randomly in a selected server type.

TABLE II. PERFORMANCE OF THE PROPOSED SOLUTION W.R.T. LOWER BOUND COST AND AVERAGE NUMBER OF VM COPIES

# of original VMs	Performance w.r.t Lower bound	average # of VM copies
200	1.13	1.33
500	1.14	1.32
1000	1.10	1.29
2000	1.14	1.31
4000	1.16	1.30
5000	1.10	1.35

Table II shows the relative performance of EVRP with respect to the derived lower bound on the total energy cost. There are two reasons behind the difference between the result of EVRP and the lower bound: i) imperfection of the algorithm, and ii) constant power consumption of the servers (independent from their utilization) and effect of the distributed resources in the datacenter. Considering the utilization of the servers, we can say that even with 90% utilization, the total energy cost of the VM placement solution is greater than the lower bound by ~10%. The average number of VM copies on the final solution of the EVRP is also shown in this table. This value is very small with respect to the average L_i for VMs which is 3. This shows that the EVRP does not make more than one copy of a VM unless it is beneficial for the energy cost of the system.

The effect of different L_i values is reported in Figure 4. In this figure the normalized total energy cost of the VM placement solution by using EVRP for different L_i values are shown. As can be seen, the difference between EVRP and a version of EVRP that restricts the number of VM copies to 2 is 4% (average). This shows that the idea of using multiple copies of VM is effective even if the number of these copies is limited to 2 for big VMs. This difference for a version of EVRP that considers at most 10 copies of VM for a VM with the biggest CPU cycle requirement is 3% (average). Based on our previous work [19], we expect that using more than a limited number of copies of a VM affect the QoS of the users even if the memory BW of each copy is equal to the memory BW of the original VM. More precisely, each copy of the VM should be able to service its share of requests and satisfy the performance requirements (response time or throughput constraint). Decreasing the CPU cycle of each VM copy limits the VM copy to a certain percentage of the requests until a case that a VM copy cannot even satisfy the performance constraint for one request at a time. This issue can be addressed by considering a lower bound on the CPU cycle requirement of each VM copy.

Figure 5 shows the average run-time of the EVRP, baseline, and mPP methods for different number of VMs. Note that VM placement algorithm is called only a few times in each charge cycle (one hour in Amazon EC2 service [22]), e.g. 2-3 times per hour. Also to reduce the time complexity of the EVRP algorithm in case of bigger number of VMs, we can use a partitioning algorithm to assign a set of VMs to a cluster and then apply EVRP in each cluster in parallel.

Figure 6 shows the average utilization of the servers for different P^p/P^0 and for different VM placement methods. As can be seen, the utilization level increases when P^p/P^0 decreases. Smaller value for P^p/P^0 means that the server is less energy-proportional. For these cases, if a server is turned ON, we will try to utilize it as much as possible. This behavior is seen for all of these algorithms. Moreover, it can be seen that increasing maximum L_i, increases the utilization level in average. This behavior is also expected because by increasing maximum L_i, we have more chance to get smaller VMs to fill the servers and avoid under-utilized servers.

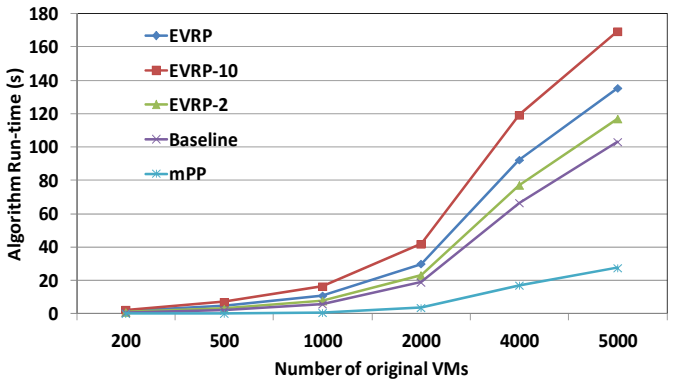

Figure 5. Run-time of EVRP for different number of VMs on 2.4GHZ E6600 server with 3GB of RAM from Intel

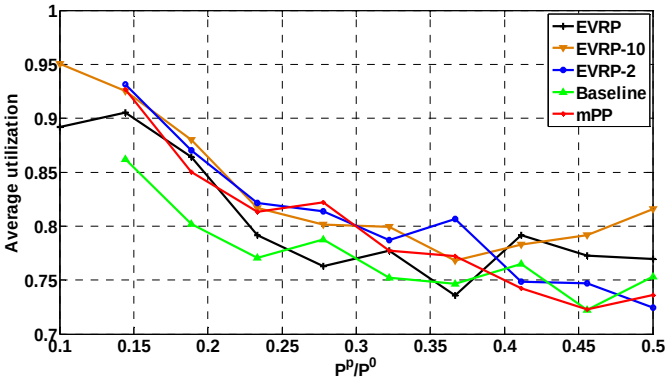

Figure 6. Ratio of expected percentage of the response time constraint's violation to the maximum allowed percentage of violation

VII. Conclusion

We presented an approach to generate multiple copies of VMs without sacrificing the QoS. An algorithm based on dynamic programming and local search was provided to determine the number of VM copies, and then place them on the servers to minimize the total energy cost in the cloud computing system. This approach reduces the energy cost by up to 20% with respect to prior VM placement techniques.

The proposed solution provides a flexible method to increase the energy efficiency of the cloud computing system or even increase the resource availability in the datacenter. Cloud provider can decide how to service VMs with big processing resource requirements and how to distribute their requests among the servers to maximize the energy efficiency.

To guarantee QoS for each VM, we only considered fixed memory BW requirement and we added a limitation on the number of VM copies. For future work, it is possible to consider that if a VM is copied, we should increase the total processing requirement by a factor. Moreover, other resources such as communication (network I/O) resources and secondary storage can be considered in this decision making. Moreover, different methods should be provisioned for cooperation and consistency between different VM copies and failure recovery.

References

[1] L. A. Barroso and U. Hölzle, "The Case for Energy-Proportional Computing," IEEE Computer, 2007.

[2] L. A. Barroso and U. Hölzle, The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines, Morgan & Claypool Publishers, 2009.

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica and M. Zaharia, "A view of cloud computing," Commun ACM 53(4), pp. 50-58.

[4] R. Buyya, "Market-oriented cloud computing: Vision, hype, and reality of delivering computing as the 5th utility," in the proc. of the 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, CCGRID 2009, May 18, 2009.

[5] R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang and X. Zhu, "No "power" struggles: Coordinated multi-level power management for the datacenter," ACM SIGPLAN Notices 43(3), pp. 48-59. 2008.

[6] S. Srikantaiah, A. Kansal, and F. Zhao, "Energy aware consolidation for cloud computing," In proc. of the 2008 conference on Power aware computing and systems (HotPower'08). 2008.

[7] X. Wang and Y. Wang, "Co-con: Coordinated control of power and application performance for virtualized server clusters," in proc. of the 17th IEEE International Workshop on Quality of Service (IWQoS). 2009.

[8] C. Tang, M. Steinder, M. Spreitzer and G. Pacifici, "A scalable application placement controller for enterprise datacenters," In proc. of the 16th International World Wide Web Conference, WWW2007, May 2007.

[9] F. Hermenier, X. Lorca, J. Menaud, G. Muller, and J. Lawall, "Entropy: a consolidation manager for clusters," In proc. of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments (VEE '09).

[10] N. Bobroff, A. Kochut, K. Beaty, "Dynamic Placement of Virtual Machines for Managing SLA Violations," In proc. of the 10th IFIP/IEEE International Symposium on Integrated Network Management, 2007. IM '07, May 2007.

[11] A. Verrna, P. Ahuja and A. Neogi, "pMapper: Power and migration cost aware application placement in virtualized systems," In proc. of the 9th ACM/IFIP/USENIX International Middleware Conference. 2008.

[12] S. Srikantaiah, A. Kansal, and F. Zhao, "Energy aware consolidation for cloud computing," In the proc. of the workshop on Power Aware Computing and Systems (HotPower '08), December 2008.

[13] Z. Liu, M. S. Squillante and J. L. Wolf, "On maximizing service-level-agreement profits," In proc. of the 3rd ACM Conference on E-Commerce, 2001.

[14] H. Goudarzi, M. Ghasemazar and M. Pedram, "SLA-based Optimization of Power and Migration Cost in Cloud Computing," in proc. of the 12th IEEE/ACM International Symposium on Cluster Computing and the Grid, CCGRID 2012, May 2012.

[15] L. Zhang and D. Ardagna, "SLA based profit optimization in autonomic computing systems," In proc. of the 2nd Int. Conf. on Service Oriented Computing, November 2004.

[16] D. Ardagna, B. Panicucci, M. Trubian, L. Zhang, "Energy-Aware Autonomic Resource Allocation in Multi-Tier Virtualized Environments," IEEE Transactions on Services Computing, 2010.

[17] A. Chandra, W. Gongt and P. Shenoy, "Dynamic resource allocation for shared clusters using online measurements," in the proc. of the ACM SIGMETRICS 2003.

[18] H. Goudarzi and M. Pedram, "Maximizing profit in the cloud computing system via resource allocation," in proc. of the international workhop on Datacenter Performance, June 2011.

[19] H. Goudarzi and M. Pedram, "Multi-dimensional SLA-based resource allocation for multi-tier cloud computing systems," in proc. of the 4th IEEE International confernece on cloud computing, July 2011.

[20] S. Martello and P. Toth, Knapsack Problems: Algorithms and Computer Implementations. , Wiley, 1990.

[21] http://ark.intel.com/

[22] http://aws.amazon.com/ec2/pricing/