Dynamic Virtual Machine Placement for Cloud Computing Environments

Xinying Zheng   Yu Cai
Michigan Technological University
Houghton, MI 49931
{zxying, cai}@mtu.edu

***Abstract*— With the increasing application of large scale cloud computing platforms, how to place virtual machine (VM) requests into available computing servers to reduce energy consumption has become a hot research subject. However, the current VM placement approaches are still not effective for live migrations with dynamic characters. In this paper, we proposed a dynamic VM placement scheme for energy efficient resource allocation in a cloud platform. Our dynamic VM placement scheme supports VM requests scheduling and live migration to minimize the number of active nodes in order to save the overall energy in a virtualized data center. Specifically, the proposed VM placement scheme is built on a statistical mathematic framework, and it incorporates all the virtualization overheads in the dynamic migration process. In addition, our scheme considers other important factors in related to power consumption, and it is ready to be extended with more considerations on users demand. We conduct extensive evaluations based on HPC jobs in a simulated environment. The results prove the effectiveness of our scheme.**

***Index Terms*— Cloud computing, virtual machine, dynamic consolidation, energy efficiency**

I. Introduction

Cloud computing, a new computing platform in which users can acquire and release the resources on demand from a Web browser, becomes one of the most explosively expanding technologies in the computing industry today. Therefore, the number and the scale of Cloud service providers have greatly increased. Amazon Elastic Compute Cloud (EC2) [1], IBM Blue Cloud [2] and Microsoft Live Mesh [3] are some of the major players. However, more data centers means more energy supply, more network usage, and causes increased heat dissipation, reduced computational density, and higher operating costs. It places a heavy burden on both environment and energy resources. It is estimated that servers consume 0.5 percent of the world total electricity, which is projected to quadruple by 2020 if current trend continues [4]. A recent IDC report estimated the worldwide spending on enterprise power and cooling to be more than $30 billion and likely to even surpass spending on new server hardware [5].

Recently, with the rapid development of virtualization technology, such as VMware [6], Xen [7], more and more data centers use this technology to build new generation data center architecture to support cloud computing [8]. Virtualization provides a method for on demand migration and dynamic allocation of these virtual machines. It allows users to achieve the same level of performance in a more flexible and secure way. It also enables workloads to consolidate to less physical servers, thus reducing overall datacenter power consumption.

The use of workload consolidation to vacate physical server nodes to improve system efficiency has been already presented in different works. A key issue in workload consolidation is to map the VMs to physical machines (PMs) [9]. Many previous works have formulated the VM mapping problem as a multi-dimensional bin-packing problem. Each dimension represents a particularly resource type of a VM request, the goal is to use as less bin as possible to fulfill all the VM requests [10][11]. The problem is NP-hard and can be solved by some heuristic methods such as first-fit or best-fit, however, those methods neglect the dynamic behavior of workload. For example, as shown in Figure 1, three jobs are currently running on two PMs. When two new jobs arrive sequentially, both first-fit and best-fit require an additional PM. However, if we firstly migrate VM_1 from PM_1 to PM_2, the new arrival jobs can be both allocated to $PM_1, while PM_3$ can remain off to save energy. On the other hand, in static consolidation strategies, when any job finishes its execution and departs the system, it releases the occupied resource so the workload concentration will be violated. Moreover, the use of consolidation strategies must also consider additional factors that are of utmost importance for data centers, such as Quality of Service (QoS), reliability in addition to energy consumption. The overheads caused by VM consolidation and migration need to be investigated.

In this paper, we propose a new dynamic VM placement scheme that can dynamically and effectively map the VM requests to PMs while saving energy. We construct a VM/PM mapping probability matrix, in which each VM request is assigned with a probability running on a specific PM. The VM/PM mapping probability matrix takes into account of resource requirements, virtualization overhead, power efficiency as well as server reliability. Our scheme then decides where to execute a new job, and whether to move existing jobs in order to improve global system efficiency. Furthermore, the proposed scheme is able to extend with more considerations in the light of users' demand. This paper discusses the entire proposed scheme and evaluates its effectiveness via extensive simulations.

The rest of this paper is organized as follows: section II reviews related work; section III describes the mathematical

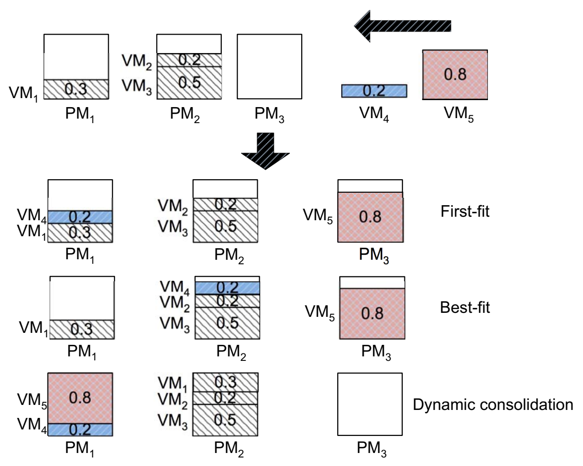


Fig. 1. Dynamic consolidation *V.S.* static consolidation.

framework of our dynamic VM placement scheme, each factor is presented in detail on how they cooperate together to lower power consumptions; section IV details the workload prediction and spare server control; we perform extensive simulations in section V; the last section concludes the paper.

II. Related Work

There is an expansion in research on energy efficiency in large scale data center or server clusters in the past few years. In this section, we only review the work related to VM management and cloud computing since they are more closely related to this work.

One most important technology that make cloud computing possible is the use of virtualization [7][12] [13]. Virtualization allows consolidation of a number of smaller workloads into partitions of a larger physical server, while the user achieves the same level of performance and security at a lower management cost and possibly lower hardware cost [11].

A significant amount of works are focused on VM scheduling and consolidation planning. The goal is to map VMs to least possible PMs without degradation of performance in a data center or a server cluster. These efforts can be divided into two categories: static VM consolidation and dynamic VM consolidation. Under static consolidation, the VM management problem is often formulated as N-dimensional set packing or N-dimensional bin packing problem, both of which have been widely studied with good approximate solutions available [14][15][16]. However, VM loads often change over time, it is not sufficient to make good initial placement choices only using static consolidation approach. It is necessary to dynamically alter placements as conditions change in a data center [17].

The technique of live migration that allows for reallocation of an executing operating system between two physical machines without significant interruption, makes dynamic VM consolidation possible [18][12]. In [8], the authors proposed a new consolidation approach in a homogeneous cluster environment that considered both VMs allocation and VMs migrations. Bo Li [19] enables live placement of application dynamically with consideration of energy efficiency in a cloud platform. Another group of researchers presented a planning tool called ReCon to recommend dynamic consolidations in a large multi-cluster data center [20]. However, these works were either formulated in a homogeneous platform or did not treat live migration overhead carefully. Our work is built on a heterogeneous platform, and has taken into considerations of all the live migration overhead and system reliability in addition to energy efficiency. In [21], the authors proposed a score-based live migration mechanism that carefully addressed virtualization overheads. However, in their work, the active number of physical servers did not depend on the dynamic VM mapping results, but depended on two workload intensity thresholds, which will not lead to the most energy savings. In our work, the dynamic VM mapping scheme and spare servers will together determine the total number of active servers, from one hand, the dynamic VM placement scheme will minimize the number of servers, from the other hand, the spare server number will be determined by workload intensity: when the system encounters less work intensity, few spare servers are needed; when the system encounters heavy workload, more spare servers are kept alive to ensure QoS. Therefore, our work will effectively reduce the system power consumption through dynamic consolidation scheme and also be capable of dealing with workload spike.

There are also some researchers made efforts to reduce the power consumption or computational cost in a cloud platform without applying dynamic consolidation. In [22], a GreenCloud architecture is proposed, which reported significant energy saving in cloud computing environment. In [23], the authors proposed a dynamic load distribution policy that addressed all electricity-related costs as well as transient cooling effects in a cloud platform. Michele Mazzucco [24] formulated a queue model to maximize the average revenue for the cloud providers. Our dynamic VM mapping scheme currently does not include cost or revenue considerations, but it is ready and easy to be extended in the future.

III. Statistical Dynamic Virtual Machine Migration

A. Problem Statement

The primary goal of VM management in a visualized data center is to minimize the PMs needed for all the VM requests. Mapping a VM correctly into PMs requires knowing the capacity of each PM and the resource requirements of the VM. It must also take into accounts VM migration overheads, reliability of PMs, QoS and the dynamic behavior of the workload. In this paper, we propose a statistical dynamic VM placement strategy which has been proved as a more effective method for VM management in a cloud environment. Our consolidation strategy is designed to find an energy efficient VM/PM mappings. A new dynamic VM placement process can be triggered by three different kinds of events: new VM arrival, VM departure, and the changes of

TABLE I
NOTATIONS

Notation	Definition
N	Number of VMs running in the system
M	Number of active PMs
p_{ij}	The joint probability of hosting VM i in PM j
$p^{xxx}ij$	The probability of hosting VM i in PM j with only consideration of condition xxx
	$xxx = res$:resource requirement $xxx = vir$: virtualization overhead
	$xxx = rel$: server reliability, $xxx = eff$: energy efficiency
R_i	resource requirement vector of VM i. $\lvert R_i\rvert = K+1$
R^{MIN}	minimum resource requirement vector. $\lvert R^{MIN}\rvert = K$
C_j^{max}	maximum resource capacity vector of PM j.
C_j	current resource occupation vector of PM j. $\lvert C_j\rvert = K$
T_i^{re}	remaining runtime of VM i
T_i^{cre}	VM creation time
T_i^{mig}	VM migration time
U_j	resource utilization (%) of PM j
U_j^{MIN}	resource utilization (%) with one minimum VM hosted in PM j
$power_j$	per VM power consumption of PM j
eff_j	power efficiency parameter of PM j
MIG_{round}	migration round limit.
$MIG_{threshold}$	migration threshold.
$n_{arrival}(t, t+T)$	number of VMs arrival in the next control period T
$n_{departure}(t, t+T)$	number of VMs departure in the next control period T
$n_{spare}(t, t+T)$	number of spare PMs in the next control period T
$n_{idle}(t)$	number of non-idle PMs at time t
$n_{AVE}(t)$	average number of PM required by a VM until time t

PM reliability. Then, the best VM/PM mapping is determined with the help of a probability matrix that is constructed by considering resource requirements, virtualization overheads, reliability and energy efficiency. We will explain the whole framework in detail in the rest of this section.

For better understanding of our model, Table I summarizes the notations and definitions which will be used throughout this paper.

B. Transition Probability Matrix Formulation

$$\begin{array}{c|cccc} & VM_1 & VM_2 & \cdots & VM_N \\ \hline PM_1 & p_{11} & p_{12} & \cdots & p_{1N} \\ PM_2 & p_{21} & p_{12} & \cdots & p_{1N} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ PM_M & p_{M1} & p_{M2} & \cdots & p_{MN} \end{array} \tag{1}$$

The probability matrix consists of M rows and N columns. Each row represents an active PM in the system, each column represents a VM that is currently running in the system. The elements in the matrix p_{ij} is the probability of hosting a VM i in a specific PM j, which is a joint probability with several constraints. There are several issues to be considered about VM allocation and migration, our current work takes into considerations of the resource requirement, the VM migration overhead, the physical server reliability and power efficiency. Hence, $p_{ij} = p_{ij}^{res} * p_{ij}^{vir} * p_{ij}^{rel} * p_{ij}^{eff}$. Since the p_{ij} is a joint probability, it is easy to be extended to accommodate other constraints in the light of users demand. In following contents we will explain the different issues in detail in our dynamic consolidation strategy.

1) Resource requirements: Each VM has a specific resource demand from PMs, such as number of CPUs, memory size or disc space. We first check if a VM can be hosted in the PMs. We define a VM request i as a $K+1$ dimensional vector: $|R_i| = K+1$. The first K components represent the resource demands of a VM. The last component is the estimated running time of the VM request. The resource capacity of a PM j is a K dimensional vector: $|C_j^{max}| = K$, each component $C_j^{max}(k)(for\ k = 1,2,...K)$ represents the maximum capacity of resource type k on PM j. A K dimensional vector C_j represents the current resource occupations of PM j. We define p_{ij}^{res} is the probability of VM i hosted in PM j by only considering the resource requirement:

$$p_{ij}^{res} = \begin{cases} 1; & \forall\ R_i(k) + C_j(k) \leqslant C_j^{max} \\ 0 & otherwise \end{cases} \tag{2}$$

The rationale of the above definition is quit straightforward, if there is sufficient resource for VM i hosted in PM j, the probability $p_{ij}^{res} = 1$, otherwise, $p_{ij}^{res} = 0$.

2) Virtualization overhead: We carefully consider two virtualization overheads in our framework. One is the time to create a VM, we refer it to as creation overhead: T^{cre}, the other is time when a VM migrating from one PM to another, we refer it to as T^{mig}. Furthermore, we also consider the remaining execution time of a VM request i requested by the users while they are submitting their jobs, reminds the VM request vector: $R_i(K+1)$. The probability of VM i hosted in PM j while only considering the virtualization overhead is defined as:

$$p_{ij}^{vir} = \begin{cases} 1; & if\ VM\ i\ hosted\ in\ PM\ j \\ \left(\dfrac{T_i^{re} - T^{cre} - T^{mig}}{T_i^{re}}\right)^2; & if\ T_i^{re} - T^{cre} - T^{mig} \geqslant 0 \\ 0. & Otherwise \end{cases} \tag{3}$$

In this definition, if the VM i is already hosted in the PM j, with no VM creation or migration overheads involved, the probability of VM i hosted in PM j is 1by only considering the migration overhead. Otherwise, we calculate the VM creation time and the VM migration time, compare it with the remaining running time. We use a quadratic equation to illustrate the penalty of virtualization overheads, the probability p_{ij}^{mig} will decrease faster as the remaining time decreases, in this case, small probability is assigned to the VM i with small remaining time, because it will finish soon and release corresponding resources, there is less need for migration. If $T_i^{re} - T^{cre} - T^{mig} = 0$, which means there is no chance for VM i migrating to PM j because of insufficient migration time, so $p_{ij}^{vir} = 0$.

3) Server reliability: We also consider server reliability in the dynamic consolidation process. Each physical machine is given a probability of reliability p_j^{rel} according to its life time, chance of failure and so on. The higher the probability is, the more reliable of the physical machine is. We use this reliability probability of the physical machine j as the probability of VM i hosted in PM j while only considering the reliability issue, so $p_{ij}^{rel} = p_j^{rel}$. If a physical machine fails, all the VMs that are running on it will be reallocated.

4) Energy efficiency: The primary goal of our scheme is to consolidate the VMs to as less PMs as possible, in other words, we want to ensure each PM is fully or nearly fully utilized to improve system efficiency. A nonuniform partition strategy is proposed to derive the p_{ij}^{eff}. We firstly evaluate the resource utilizations of each physical machine and partition it into W_j levels. After that, the probability p_{ij}^{eff} is assigned according to resource utilization levels and energy efficiency parameter of the PM j. The detailed process and rationale are explained in detail as following.

It is important to consider multiple resource types instead of a single resource type to better evaluate the resource utilization of a physical machine. For example, if 100% CPU of a PM is utilized while only 20%memory is utilized. In this case, no more jobs can be allocated in the physical machine, wasting 80% of memory.

In our work, the resource utilization of the physical machine j is a joint product of several resource utilizations: $U_j = \prod \frac{C_j(k)}{C_j^{MAX}(k)}$, for $k \in \{1,2,3..K\}$, each $\frac{C_j(k)}{C_j^{MAX}(k)}$ represents the utilization of resource type k. Larger value means better utilizations. We define a minimum resource requirement of a VM request as $|R^{MIN}| = K$, each component $R^{MIN}(k)$ represents the requirement of resource type k of a physical machine. If there is only one VM with minimal resource requirements allocated on the PM j. The resource utilizations of PM j is $U_j^{MIN} = \prod \frac{R^{MIN}(k)}{C_j^{MAX}(k)}$, for $k \in \{1,2,3..K\}$. If two VMs with minimal resource requirements are running on the PM j, the resource utilization is $U_j = 2^k \frac{R^{MIN}(k)}{C_j^{MAX}(k)}$. Assuming the PM j has sufficient resources for a maximum number of W_j VMs, the maximum resource utilization of PM j is $U_j = W_j^k \frac{R^{MIN}(k)}{C_j^{MAX}(k)}$. For each PM j, we partition the resource utilization by the number of VMs that can be hosted on it and category it into $W_j + 1$ levels as shown in equation 4. Note, we formulate the problem in a non-homogeneous server system, so each server may have different computation capacity, that is why we separately partition the resource utilizations into different groups for different PMs. If all the PMs are identical nodes: $W_1 = W_2 = \cdots = W_j = \cdots = W_M$.

$$\begin{aligned}
L_0 &= [0, U_j^{MIN}) \\
L_1 &= [U_j^{MIN}, 2^k U_j^{MIN}) \\
L_2 &= [2^k U_j^{MIN}, 3^k U_j^{MIN}) \\
&\vdots \\
L_{w_j} &= [(w_j)^k U_j^{MIN}, (w_j+1)^k U_j^{MIN}) \\
&\vdots \\
L_{W_j - 1} &= [(W_j - 1)^k U_j^{MIN}, W_j^k U_j^{MIN}) \\
L_{W_j} &= [W_j{}^k U_j^{MIN}, 1]
\end{aligned} \tag{4}$$

We also define a relative power efficiency parameter to balance the power efficiency in different server nodes: $eff_j = \frac{min:\{power_j\}}{power_j}$(for $j = 1,2,\ldots,M$), where $power_j$ is the active power consumption of PM j divided by W_j, in other words, it is the per VM power consumption of PM j. The PM j with minimal per VM power consumption will have a relative power efficiency parameter equal to 1. The PM with higher per unit power consumption will be assigned with a smaller value. After that, the energy efficiency probability is defined to be proportional to its utilization levels and its relative power efficiency as shown in equation (5):

$$p_{ij}^{eff} = \frac{w_j}{W_j} \cdot eff_j, \qquad wj \in \{1,2,....W_j\} \tag{5}$$

In the above definition: level L_0 means that there is no VM hosted in the PM j, so the PM j is completely idle (There is no possibility that the resource occupation R_j^{occ} is less than the minimal VM resource requirement R_j^{MINocc}.). L_1 means that there is no more than one VM hosted in PM j. As the level increases, the utilization of PM j increases. At the last level, PM j is fully utilized or nearly fully utilized, thus no further VM requests can be accepted, for which we consider that the PMj has achieve its maximal energy efficiency. By partitioning the resource occupation, VM i has higher probability to be hosted in the PM j with higher utilization level and energy efficiency PMs, which will eventually minimize the total active physical machines.

C. Dynamic consolidation process

We firstly identify which event causes a new VM migration process. If a new VM request arrives, we only calculate the probability in the new VM column and allocate it to the PM with the highest probability, and starts our dynamic VM migration process; if a finished job departs from the system, the dynamic VM migration process starts immediately; if a PM fails, all the VMs hosted in that PM will be treated as new VM requests.

Algorithm 1 Dynamic VM migration algorithm

```
Transition probability matrix $P$ initialization
for each column
  Normalize the $P$ by dividing the probability of the current hosted VM and obtain normalized matrix $D$
end for
While there is values larger than $MIG_{threshold}$ in $D$ and migration round less than $MIG_{round}$ do
    1. Select the largest value in the matrix $d_{ij}$
    2. Move VM $j$ from the current PM $m$ to PM $j$
    3. Update matrix $P$ in rows $m$ and $j$
    4. Update matrix $D$ in rows $m$, $j$ and column $i$.
End while
```

The dynamic migration algorithm is shown in algorithm1. One VM migration round consists of three steps: probability matrix construction, probability matrix normalization and VM migration. In the first step, we calculate the joint probability p_{ij} by considering resource requirements, VM migration overhead, energy efficiency and server reliability. We then check if there is a better VM/PM mapping compared to the current one. We divide each p_{ij} with $p_{i(PM_{current})}$ (which is the probability the VM i is currently allocated on) in each column to obtain a normalized probability matrix. Numbers in the matrix with value greater then 1, indicate efficiency improvement. Numbers which are less than 1 correspond to degradation while 1 indicates that the VM is indicates that the VM is currently hosted in the PM. In the last step, we select the largest value in the normalized probability matrix, and move corresponding VM to the new PM, having the VM/PM mapping updated. After that, we release the PM resources on which the VM moves from, update the PM resource occupied on which the VM moves to. Here we finished the first round, in the next VM migration round, we only need to update the corresponding PM rows in the last migration process instead of calculate all the probabilities, and start the new VM migration process.

probability matrix

$$\begin{array}{c|ccccc} & VM_1 & VM_2 & VM_3 & VM_4 & VM_5 \\ \hline PM_1 & 0.60 & 0.70 & 0.90 & 0.60 & 0.65 \\ PM_2 & 0.80 & 0.90 & 0.40 & 0.78 & 0.55 \\ PM_3 & 0.50 & 0.80 & 0.91 & 0.75 & 0.80 \end{array}$$

To better understand our dynamic VM migration strategy, we use an example to illustrate the whole process. In the example, there are 5 VMs are currently running on three PMs, in which VM_1 is running on PM_2, VM_2 on PM_1, VM_3 on PM_1, VM_4 on PM_3 and VM_5 on PM_3. We first obtain a probability matrix as shown in the probability matrix, each element represents the probability of VM i hosted on PM j. We then normalize the probability matrix in each column to obtain the normalized matrix. For example, in VM_1 column, each element is divided by 0.8, since VM_1 is currently running on PM_2, same for the rest of the columns. In the normalized matrix below, we observe that 1.28 is the largest value, so we migrate VM_2 to PM_2, release the resource of PM_1, refresh the PM_1 and PM_2 rows in the probability matrix and be prepared for the next migration round.

normalized matrix

$$\begin{array}{c|ccccc} & VM_1 & VM_2 & VM_3 & VM_4 & VM_5 \\ \hline PM_1 & 0.75 & 1.00 & 1.00 & 0.80 & 0.81 \\ PM_2 & 1.00 & 1.28 & 0.44 & 1.04 & 0.68 \\ PM_3 & 0.63 & 1.14 & 1.01 & 1 & 1.00 \end{array}$$

To minimize unnecessary migrations, we set two thresholds to restrict the dynamic migrations. One is migration times MIG_{round}, which is the number of migration rounds. When the migration rounds reach the limits, the dynamic migration process stops immediately. The other is migration threshold $MIG_{threshold}$, which ensures that only those migrations resulting in improvement will be counted. For example, if we set $MIG_{threshold} = 1.05$, the migration process will stop if there is no number larger than 1.05 in the normalized matrix. We do not utilize in-depth search in the migration process to save computation time and space, however, each migration round will bring a more efficient and reliable VM/PM mapping until the migration terminated.

IV. Workload Prediction and Spare Server Controlling

In addition to dynamic VM consolidation, another important decision is to determine the amount of active physical servers in the system. The number of active physical machine should be large enough to handle the unexpected workload spike and avoid performance degradation. It should also be as less as possible to achieve energy efficiency in the system. We periodically determine the active PMs from two aspects. In a time slot T, the total number of active PMs $N_{ac}(t,t+T)$ is the summation of non-idle PMs $N_{idle}(t)$ and spare servers $N_{spare}(t,t+T)$. The non-idle PMs is the number of PMs that hosted VM requests, which can be easily derived. The spare server $N_{spare}(t,t+T)$ is determined by modeling the incoming VM requests as a non-homogeneous Poisson process.

In this paper, the incoming VM requests is formulated as a non-homogeneous Poisson process, whose rate $\lambda(t)$ varies over time. The accumulative intensity function in a time slot T can be defined as:

$$\Lambda(t,t+T) = \int_{t}^{t+T} \lambda(t)dt \tag{6}$$

We estimate the cumulative VM requests arrival rate function $\widetilde{\Lambda(t,t+T)}$ using a non parametric estimation method in [25]. The probability of no more than n requests arrive in a control period T can be obtained:

$$P(\widetilde{\Lambda(t,t+T)} > n) = 1 - P(\widetilde{\Lambda(t,t+T)} \leqslant n) \tag{7}$$

We use the above equation to estimate the number of arrival VM requests in the next time slot, and also ensure $P(\widetilde{\Lambda(t,t+T)} > n)$ less than a threshold for QoS. In this way, enough spare servers will be in an active mode waiting to

TABLE II

DATA CENTER PARAMETER SETTINGS

Nodes	Fast	Slow
Number	25	75
VM creation time(seconds)	30	40
VM migration time (seconds)	40	45
ON/OFF overhead (seconds)	50	55
Number of processors	2	2
Cores per processor	4	2
Memory (G)	8	4
Active power consumption (W)	400	300
Idle power consumption (W)	240	180

serve the incoming requests. In this paper, we ensure that less than 5% of VM requests have to wait in the queue because of insufficient PMs. So the estimated number of arrival VMs $n_{arrival}(t,t+T)$ is determined by $P(\Lambda(T) > n_{arrival}) \leqslant 0.05$.

We define another parameter $N_{Ave}(t)$, which is the average number of PM required by a VM request. This number can be computed by the number of VM requests running in the system divide the non-idle physical servers $N_{nidle}(t)$. To best estimated this parameter, $N_{Ave}(t)$ is dynamically updated after each dynamic VM migration process. The spare server is determined by the following equation:

$$N_{spare}(t,t+T) = \begin{cases} 0; \\ (if\ n_{arrival}(t,t+T) - n_{departure}(t,t+T) \leqslant 0) \\ \dfrac{n_{arrival}(t,t+T) - n_{departure}(t,t+T)}{N_{Ave}(t)} \\ (Otherwise) \end{cases} \tag{8}$$

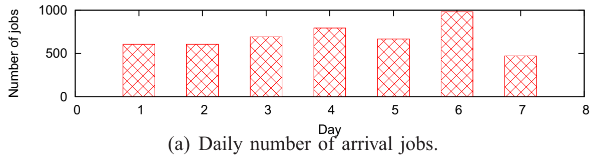
In equation(8), $n_{departure}(t,t+T)$ is the number of VM requests that will finish their execution and depart from the system in the next control period. It can be easily derived, since each VM request is submitted with an estimated running time. If more VM requests depart the system, there is no need to keep spare servers, the number of active server in the current time $N_{nidle}(t)$ is sufficient to handle the incoming request, idle server will be turned off during the dynamic consolidation process. However, we will have no less than $N_{nidle}(t) + \dfrac{n_{arrival}(t,t+T) - n_{departure}(t,t+T)}{N_{Ave}(t)}$ servers in the active mode; On the contrary, if there are more VMs arriving the system, we will keep $N_{spare}(t,t+T)$ spare servers active.

V. EVALUATION

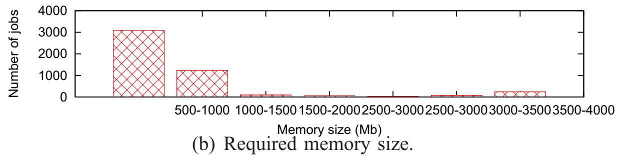
A. Parameters setting

We built a simulator to evaluate our dynamic VM migration scheme. It takes workload trace as input and outputs the performance and the power consumption. The datacenter is configured to have 100 nodes, including 25 fast nodes and 75 slow nodes according to their virtualization overheads and computation capacities. The detailed information and parameter settings of the virtualized data center is shown in table II.

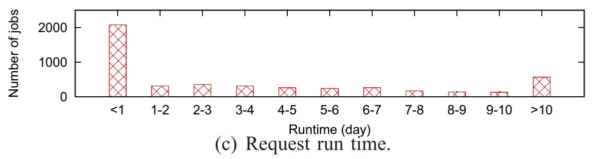
Because lack of Cloud Computing traces, we use a slightly modified trace of jobs submitted to an HPC cluster available



(a) Daily number of arrival jobs.



(b) Required memory size.



(c) Request run time.

Fig. 2. Workload characteristics.

from the ParallelWorkloads Archive [26]. This trace contains approximately ten months (August 2004 through May 2005) of data. The trace, LPC Log, contains a record for each job serviced by the cluster, with each record containing a job number, submitted time, actual run-time, and maximum number of cores and amount of memory used. We extracted a week from this trace, and filter out the canceled jobs, jobs with small memory requirements, then use it as the workload for all the simulations discussed below.

Figure 2 (a) shows the number of arrival jobs per day. This week long workload contains 4574 jobs, with a peak demand of 982 VM requests per day. We only consider two resource types in the simulation: CPU and memory. We have normalized the memory required by each job by equally dividing its number of cores required. So each VM request requires a single core, a specific memory size with an estimate of its run-time. The required memory and runtime for the week long trace are shown in Figure 2 (b)(c). We noticed that most jobs require the memories of less than 1GB. There are 2077 jobs with a runtime less than a day. The workload indicates that jobs arrive and leave frequently, which makes dynamic consolidation necessary for system efficiency purpose.

We compare our proposed VM consolidation scheme with two static schemes: the first is first-fit scheme, in which the new arrival VM request will be placed to the first PM with available computation resources; the second is best-fit scheme, in which the new arrival VM request will be placed to the PM that can achieve its maximum utilization. Figure 3 shows the number of active servers every hour in a week of testing period. As we can see, the dynamic consolidation scheme can effectively reduce the number of active running servers in the cloud without sacrificing VM QoS requirements. Figure 4 and Figure 5 further illustrates the saving of power consumption with the dynamic consolidation scheme compared with other two static schemes. Figure 4 is the

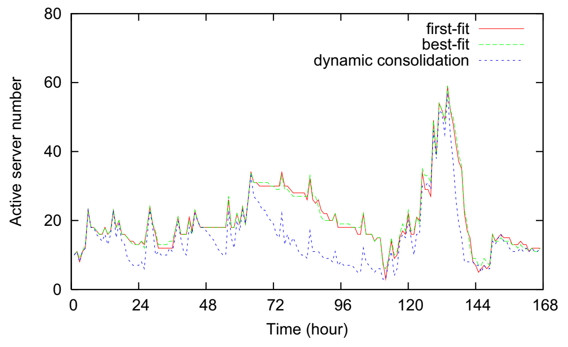


Fig. 3. Hourly active server numbers in a week.

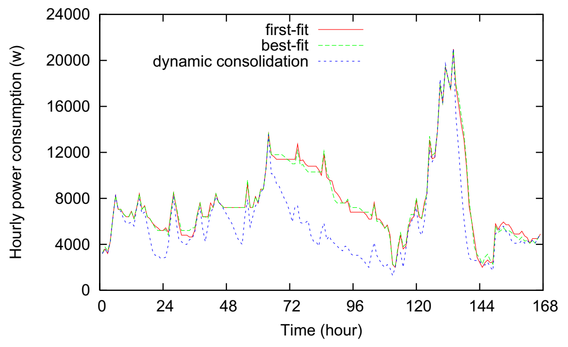


Fig. 4. Hourly power consumption in a week.

hourly power consumption in a week; and Figure 5 is the daily power consumption in a week.

VI. Conclusion and Future Work

This paper proposes a dynamic VM placement scheme in a cloud computing environment. The contributions can be summarized as follows: first, we construct a statistical framework for VM placement. A VM/PM mapping probability matrix is constructed by considering the resource requirements, virtualization overheads, server reliability and energy efficiency; second, we use a normalized probability matrix to decide how to migrate VMs in order to improve system efficiency; third, we propose a spare server strategy to determine the number of active servers in each control period. The proposed strategy can efficiently reduce the system power consumption and also be capable of dealing with workload spike; finally, our dynamic VM placement scheme is evaluated by extensive simulations. Simulation results show that our dynamic scheme can save significant energy compared to static scheme with satisfied performance.

In the future, we plan to extend our current work to a multiple geographical data center environment with electricity cost and revenue considerations. The dynamic behavior of electricity price will be formulated as an important factor in the dynamic VM migration process. In this work, VM migrations will be performed not only inside a data center but also among data centers. We believe this work will bring new insights for cloud providers.

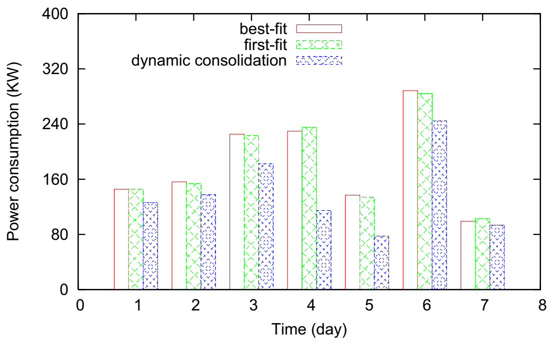


Fig. 5. Daily power consumption

References

[1] Amazon EC2. http://aws.amazon.com/ec2/.

[2] IBM Cloud. http://www.ibm.com/cloud-computing/us/en/.

[3] Microsoft Live Mesh. http://connect.microsoft.com/LiveMesh.

[4] M. Tedre, B. Chachage, and J. Faida, "Integrating environmental issues in IT education in Tanzania," *Frontiers in Education Conference, 2009. FIE '09. 39th IEEE*, pp. 1 – 7, Oct 2009.

[5] R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang, and X. Zhu, "No power struggles: Coordinated multi-level power management for the data center," in *in ASPLOS*, 2008.

[6] C. A. Waldspurger, "Memory resource management in VMware ESX server," *SIGOPS Oper. Syst. Rev.*, vol. 36, pp. 181–194, December 2002.

[7] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," in *Proceedings of the nineteenth ACM symposium on Operating systems principles*, SOSP '03, (New York, NY, USA), pp. 164–177, ACM, 2003.

[8] F. Hermenier, X. Lorca, J.-M. Menaud, G. Muller, and J. Lawall, "Entropy: a consolidation manager for clusters," in *Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*, VEE '09, (New York, NY, USA), pp. 41–50, ACM, 2009.

[9] V. Petrucci, O. Loques, and D. Mossé, "A dynamic optimization model for power and performance management of virtualized clusters," in *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, e-Energy '10, (New York, NY, USA), pp. 225–233, ACM, 2010.

[10] R. Nathuji and K. Schwan, "VirtualPower: coordinated power management in virtualized enterprise systems," in *Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles*, SOSP '07, (New York, NY, USA), pp. 265–278, ACM, 2007.

[11] A. Verma, P. Ahuja, and A. Neogi, "Power-aware dynamic placement of HPC applications," in *Proceedings of the 22nd annual international conference on Supercomputing*, ICS '08, (New York, NY, USA), pp. 175–184, ACM, 2008.

[12] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation - Volume 2*, NSDI'05, (Berkeley, CA, USA), pp. 273–286, USENIX Association, 2005.

[13] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Black-box and gray-box strategies for virtual machine migration," in *Proceedings of the 4th USENIX conference on Networked systems design &#38; implementation*, NSDI'07, (Berkeley, CA, USA), pp. 17–17, USENIX Association, 2007.

[14] J. Shahabuddin, A. Chrungoo, V. Gupta, S. Juneja, S. Kapoor, and A. Kumar, "Stream-Packing: Resource Allocation in Web Server Farms with a QoS Guarantee," in *Proceedings of the 8th International Conference on High Performance Computing*, HiPC '01, (London, UK), pp. 182–191, Springer-Verlag, 2001.

[15] W. Leinberger, G. Karypis, and V. Kumar, "Multi-Capacity Bin Packing Algorithms with Applications to Job Scheduling under Multiple Constraints," in *Proceedings of the 1999 International Conference on*

Parallel Processing, ICPP '99, (Washington, DC, USA), pp. 404–, IEEE Computer Society, 1999.

[16] M. Hoyer, K. Schröder, and W. Nebel, "Statistical static capacity management in virtualized data centers supporting fine grained QoS specification," in *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, e-Energy '10, (New York, NY, USA), pp. 51–60, ACM, 2010.

[17] C. Hyser, B. McKee, R. Gardner, and B. J. Watson, "Autonomic Virtual Machine Placement in Data Center.," in *HP Technical report, http://www.hpl.hp.com/techreports/2007/HPL-2007-189.pdf*, Feb. 2008.

[18] R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang, and X. Zhu, "No "power" struggles: coordinated multi-level power management for the data center," *SIGARCH Comput. Archit. News*, vol. 36, pp. 48–59, March 2008.

[19] B. Li, J. Li, J. Huai, T. Wo, Q. Li, and L. Zhong, "EnaCloud: An Energy-Saving Application Live Placement Approach for Cloud Computing Environments," in *Cloud Computing, 2009. CLOUD '09. IEEE International Conference on*, pp. 17 –24, sept. 2009.

[20] S. Mehta and A. Neogi, "ReCon: A tool to Recommend dynamic server Consolidation in multi-cluster data centers," in *Network Operations and Management Symposium, 2008. NOMS 2008. IEEE*, pp. 363 –370, april 2008.

[21] I. Goiri, F. Juli and, R. Nou, J. Berral, J. Guitart, and J. Torres, "Energy-Aware Scheduling in Virtualized Datacenters," in *Cluster Computing (CLUSTER), 2010 IEEE International Conference on*, pp. 58 –67, sept. 2010.

[22] L. Liu, H. Wang, X. Liu, X. Jin, W. B. He, Q. B. Wang, and Y. Chen, "GreenCloud: a new architecture for green data center," in *Proceedings of the 6th international conference industry session on Autonomic computing and communications industry session*, ICAC-INDST '09, (New York, NY, USA), pp. 29–38, ACM, 2009.

[23] K. Le, J. Zhang, J. Meng, R. Bianchini, T. D. Nguyen, and Y. Jaluria, "Reducing Electricity Cost Through Virtual Machine Placement in High Performance Computing Clouds," in *Technical report, to be appeared In Proceedings of Super Computing (SC11), November 2011*, sept. 2011.

[24] M. Mazzucco, D. Dyachuk, and R. Deters, "Maximizing Cloud Providers' Revenues via Energy Aware Allocation Policies," in *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, pp. 131 –138, july 2010.

[25] L. M. Leemis, "Nonparametric Estimation of the Cumulative Intensity Function for a Nonhomogeneous Poisson Process," *Management Science*, vol. 37, no. 7, pp. pp. 886–900, 1991.

[26] Parallel Workloads Archive. http://www.cs.huji.ac.il/labs/parallel/workload/.