Mathematical and Computer Modelling 58 (2013) 1222–1235

Contents lists available at SciVerse ScienceDirect

Mathematical and Computer Modelling

journal homepage: www.elsevier.com/locate/mcm

Energy efficient virtual machine placement algorithm with balanced and improved resource utilization in a data center

Xin Li [a], Zhuzhong Qian [a,*], Sanglu Lu [a], Jie Wu [b]

[a] State Key Laboratory for Novel Software Technology, Nanjing University, China
[b] Department of Computer and Information Sciences, Temple University, USA

ARTICLE INFO

Article history:
Received 28 September 2012
Received in revised form 11 February 2013
Accepted 12 February 2013

Keywords:
Green data center
Virtual machine placement
Balanced resource utilization
Energy efficient

ABSTRACT

Powerful data centers are the essential supporting infrastructure for mobile, ubiquitous, and cognitive computing, which are the most popular computing paradigms to utilize all kinds of physical resources and provide various services. To ensure the high quality of services, the performance and cost of a data center is a critical factor. In this paper, we investigate the issue of increasing the resource utilization of data centers to improve their performance and lower the cost. It is an efficient way to increase resource utilization via resource sharing. Technically, server virtualization provides the opportunity to share resources in data centers. However, it also introduces other problems, the primary problem being virtual machine placement (VMP), which is to choose a proper physical machine (PM) to deploy virtual machines (VMs) in runtime. We study the virtual machine placement problem with the target of minimizing the total energy consumption by the running of PMs, which is also an indication of resource utilization and the cost of a data center. Due to the multiple dimensionality of physical resources, there always exists a waste of resources, which results from the imbalanced use of multi-dimensional resources. To characterize the multi-dimensional resource usage states of PMs, we present a multi-dimensional space partition model. Based on this model, we then propose a virtual machine placement algorithm EAGLE, which can balance the utilization of multi-dimensional resources, reduce the number of running PMs, and thus lower the energy consumption. We also evaluate our proposed balanced algorithm EAGLE via extensive simulations and experiments on real traces. Experimental results show, over the long run, that EAGLE can save as much as 15% more energy than the first fit algorithm.

1. Introduction

We are entering into the new cyber-physical world, along with the development and evolution of key technologies like mobile and wireless communication, context sensing, and machine learning. Academic and industrial communities have paid significant attention to mobile, ubiquitous, and cognitive computing; remarkable features of computing are presented to the users: mobility, ubiquity, and intelligence [1,2]. To profit from these excellent features, more personalized and intelligent services can be provided, pervasively based on the dynamic environment [3,4]. However, because of the low capability of front-end devices and their mobility, a strong back-end infrastructure is needed to store and process the huge amount of data effectively, to provide appropriate services. Data centers are the facilities that support such huge data processing [5–8].

* Corresponding author. Tel.: +86 025 83686292.
E-mail addresses: lixin@dislab.nju.edu.cn (X. Li), qzz@nju.edu.cn (Z. Qian), sanglu@nju.edu.cn (S. Lu), jiewu@temple.edu (J. Wu).

0895-7177/$ – see front matter
doi:10.1016/j.mcm.2013.02.003

As the important supporting infrastructure, the performance of a data center directly affects the quality of service provided. Also, the cost of maintaining a data center composes a significant portion of the cost of providing timely service [5]. These facts make it a basic research issue: increasing the resource utilization and also improving the performance of a data center can cut down the maintenance cost, as well as increase the quality of provided service.

One key technology used to increase resource utilization is server virtualization. Virtual machines (VMs), each of which acts like a real computer with an operating system, are created on underlying physical machines (PMs). With such virtualization, the resources can be scheduled with fine-granularity, which improves resource utilization significantly. At the same time, though VMs share physical resources, each VM runs individually with a proprietary resource; this makes it possible to guarantee the quality of provided service. Server virtualization is an efficient way to increase resource utilization and improve quality of service.

The basic key issue in server virtualization is virtual machine placement (VMP), which is to select some suitable PM to deploy each newly-created VM in runtime. VMP is a primary problem, while it is a highly complex task caused by various constraints, including performance [9], scalability [10], availability [11], network [12–14], cost [15], etc.

One important concern during the virtual machine placement process is reducing energy consumption caused by the running PMs. Existing works [16,17] show the importance of the virtual machine placement problem, and we can benefit from the appropriate placement policy on energy saving. Cutting down energy consumption has significant impacts on (i) decreasing the energy cost, which composes a significant portion of the total maintenance cost of data centers, and keeps on growing since the unit price of energy rises over time, and (ii) making the data center more 'green', and sustainable, regarding energy.

One efficient way to cut down the energy consumption is to reduce the number of running PMs [16]. As different VMs require different kinds and amounts of resources (e.g., CPU, memory, disk, bandwidth, etc.), when placing VMs on PMs to meet such requirements, a PM cannot host any more VM as long as an arbitrary dimensional resource is exhausted, even when all other dimensional resources are sufficient; this is a phenomenon well-known as *bucket effects*. Thus, in those fully loaded PMs, all the unutilized resources, referred to as *resource fragments*, are wasted. So, in order to improve the resource utilization such that the number of running PMs is minimized, we have to lower the number of resource fragments and decrease their sizes, as well.

Since the resource fragments originate from the imbalanced use of resources over different dimensions, the placement of VMs on PMs should be executed in a resource-balanced manner. To achieve this, in this paper, we propose a multi-dimensional space partition model to characterize the resource usage of each PM. In this model, each dimension of the space corresponds to a one-dimensional resource, and the whole space is partitioned into three domains with distinctive features, which indicate the suitability of resource utilization for each placement task.

Based on the multi-dimensional space partition model, we further propose EAGLE, an energy efficient online VM placement algorithm with balanced resource utilization. When a new VM placement task arrives, our algorithm checks the posterior resource usage state for each feasible PM, and then chooses the most suitable PM, according to our proposed model. Compared to traditional greedy algorithms such as the first fit algorithm, which places the VM on any arbitrary feasible PM, our algorithm can utilize the resource in a more balanced manner, thus introducing fewer and smaller resource fragments, and further consuming less energy.

Our main contributions are two-fold:

- We introduce a multi-dimensional space partition model, which can be used to guide the design of the resource-balanced VM placement algorithm.
- We propose an energy efficient VM placement algorithm EAGLE, and evaluate its performance via extensive experiments, which reveal that EAGLE can save about 15% more energy, compared to the first fit algorithm.

The remainder of the paper is organized as follows. Section 2 reviews the related work. In Section 3, we introduce problem formulation and discuss its hardness. We present the multi-dimensional space partition model in Section 4 and the balanced algorithm EAGLE is proposed in Section 5. Experiment results are shown in Sections 6 and 7 concludes this paper.

2. Related work

As already noted in Section 1, the placement of virtual machines (VMs) on a cluster of physical machines (PMs) is a primary task under the emerging virtualization technology. Various factors were considered in the process of VM placement. [18] presents a high level overview of VM placement, and proposes a VM placement system architecture design, which adopts autonomic VM placement, to achieve cost savings from the better utilization of computing resources; however, it lacks a resource management policy.

Network, traffic, or bandwidth are the factors that attract lots of researchers. [13] optimizes the placement of VMs in a traffic-aware way. Traffic patterns among VMs were aligned with the communication distance between them, VMs with mutual bandwidth usage are assigned to PMs in close proximity. [10] takes network conditions into account to minimize the data transfer time consumption and maintain application performance. To achieve better performance, live migration is adopted in their work. Similarly, [12] formulates the VM consolidation with the network bandwidth constraint into a stochastic bin packing problem. The optimal solution is given, the number of PMs required is within $(1+\epsilon)(\sqrt{2}+1)$ of the

optimum for any $\epsilon > 0$. The authors of [14] solve a joint tenant placement and route selection problem by exploiting the multipath routing capability and dynamic virtual machine migration.

Availability is the concerned issue in [11], the authors define a new high-availability property for a VM. As long as there are up to k PM failures, it is guaranteed that a VM can be relocated to a non-failed PM without relocating other VMs, when a VM is marked as *k-resilient*. An application profiling technique was employed in [19] to improve resource utilization in the process of VM placement. There is a tradeoff between application performance and resource usage. The task of VM placement is widely investigated under various constraints and different objectives. However, most related works place the VMs in a static manner, which is totally different from online VM placement in this paper.

Previous works deal with various VMP problems, with emphasis on system properties, e.g. scalability, availability, etc. However, these works are founded on a primary assumption: the set of VMs is given beforehand, which always violates reality. Essentially, the online VMP problem is a process of meeting resource requirements from tenants, which is similar to the multi-dimensional vector bin packing problem, due to the dimensionality of resources.

For $d \geq 2$, the d-dimensional vector bin packing problem has no APTAS (unless $P = \text{NP}$) [20]. [21] analyzes the basic feature of multi-dimensional vector bin packing, and gives the lower bound of FFD-based variants for some special cases. [22] shows a $d^{\frac{1}{2}-\epsilon}$ hardness of approximation. [23] shows that one can get $(d+\epsilon)$-approximation in linear time, for any $\epsilon > 0$. [24] proposes a polynomial time approximation scheme for randomized instances of the multi-dimensional vector bin packing. [25] presents a framework for analyzing online bin packing algorithms, and proposes an algorithm within the framework; the algorithm has an asymptotic performance ratio, at most, 1.58889. Yao [26] shows that no algorithm running in time $o(n \log n)$ can give better than a d-approximation.

Recently, an online virtual machine placement problem has also been proposed [27,28], and they put the network as their concern. [27] studies the virtual machine placement under the consideration of providing bandwidth guarantees. The authors take the bandwidth as a constraint while placing the VMs, and the objective is also to maximize resource utilization. Different from [27,28] takes the network as the objective while placing VMs in distributed clouds, which works similarly in a single cloud. The proposed algorithms place the VMs, which communicate with each other on PMs as close as possible, to reduce the communication latency. However, in these two works, the other physical resources are generally treated as slots, as in [13]. It is very different from the scenario of our paper; we deal with resource requirements of various amounts. Networks are not modeled separately, but are treated as resources like CPU and memory; the traffic between VMs is not considered yet in our paper. We will handle the virtual machine placement problem while jointly considering the various physical resource requirement, and will consider bandwidth demands in the future work.

Li et al. [29] investigate the online virtual machine placement problem with the goal of saving energy, and they propose a balanced algorithm. However, the concept of resource leak is defined roughly, and there is no quantitative method to characterize the suitability of resource utilization. In this paper, we deal with the same online VMP problem. We present a precise model to represent the usage of multi-dimensional resources, and a balanced algorithm to improve resource utilization. Also, more sufficient experiments are carried out, and the model and algorithm are evaluated extensively. This paper has more new investigations than were offered by [29].

3. Problem statement

In this section, we provide the preliminaries of the online virtual machine placement problem, and then provide a formal definition of the problem. After that, we analyze the energy consumption formally, and prove that the online virtual machine placement problem is NP-complete.

3.1. Problem definition

Considering the data center, which owns unlimited uniform PMs, it provides the service of renting VMs, e.g. Amazon EC2. When tenants put forward their VM requests, some PM(s) in the data center will be selected to create the required VMs; this process is known as virtual machine placement. In this paper, we will not distinguish the notation of a VM request from a VM. Meanwhile, virtual machine placement will be described as *some PM is selected to host the VM*.

To formalize our problem, we make the following assumptions:

- Time is divided into time-slots with equal length Δt. There are $K(\tau)$ VMs to be placed at the τth time-slot.
- a PM consists of D-dimensional resources with the same capacity $\mathcal{C}$.

Actually, though the capacities of the D-dimensional resources are different, it is easy to normalize to be in the same range. At the same time, we make the following settings:

- We use p to identify a PM and v to identify a VM, respectively.
- $N(\tau)$ and $M(\tau)$ are used to represent the total number of PMs and VMs at the τth time-slot.
- We use $S(T)$ to represent a VM (request) sequence with T time-slots.

As learned from the assumptions and settings, we have

$$M(\tau) = \sum_{t=1}^{\tau} K(t).$$

In the τth time-slot, PMs are selected to host the $K(\tau)$ VMs; the D-dimensional resource should subject the primary resource constraint.

$$\mathcal{C} \geq \sum_{v=1}^{M(\tau)} A_{vp} * R_v^d, \quad \forall 1 \leq p \leq N(\tau),\ 1 \leq d \leq D \tag{1}$$

where $R_v^d (1 \leq d \leq D)$ represents the amount of the dth dimensional resource of the vth VM and

$$A_{vp} = \begin{cases} 1, & \text{if the } v\text{th VM is placed on the } p\text{th PM;} \\ 0, & \text{otherwise.} \end{cases}$$

The objective is to minimize energy consumption caused by running PMs. It is believed that reducing the number of PMs can lead to decreased energy consumption [16]. We will use *the number of PMs* as an indication of *energy consumption* in this paper. Section 3.2 gives the formalized interpretation.

We introduce a function $\boldsymbol{\pi}$ to indicate the placement schema. We let $\boldsymbol{\pi}(v) = p$ if the vth VM is placed on the pth PM. The online VMP problem can be formalized as:

Definition 1 (*Online Virtual Machine Placement Problem*)**.** Given VM sequence $S(T_0)$ and unlimited uniform PMs with capacity $\mathcal{C}$, find a placement function $\boldsymbol{\pi}$, subject to the primary resource constraint (shown in Eq. (1)), such that $N(\tau)$ ($\forall \tau \leq T_0$) is minimized, which means the energy consumption is minimized.

For a given $S(T_0)$, minimizing $N(T_0)$ is a typical multi-dimensional vector bin packing problem, which has been proved to be NP-complete. This indicates that the online virtual machine placement problem is an NP-complete problem. We will prove this conclusion in the next subsection.

3.2. Energy analysis

Energy can be represented as $E = P * T$, where E and P refer to *energy* and *power*, while T means time period. We have the basic equation:

$$E = \int_1^{T_0} P(t)dt = \Delta t \sum_{\tau=1}^{T_0} P(\tau).$$

Experiments (Section 6.1) show that *power* is mainly determined by the *CPU utilization*, it can be formulated as:

$$P = a * r + b$$

where r is *CPU utilization ratio*, and a and b are constants associated with the special resource configuration of PM.

Hence, the total energy consumption E can be formulated as:

$$E = \frac{a * \Delta t}{\mathcal{C}} \sum_{\tau=1}^{T_0} \sum_{v=1}^{M(\tau)} R_v + b * \Delta t \sum_{\tau=1}^{T_0} N(\tau) \tag{2}$$

where R_v refers to the CPU volume of the vth VM.

For a given VM sequence $S(T_0)$, $N(\tau)$ is the only variable. As shown in Definition 1, to minimize the energy consumption E, it is necessary to minimize the value of $N(\tau)$. Next, we prove that it is NP-complete to minimize the value of $N(\tau)$.

Theorem. *The online virtual machine placement problem is NP-complete.*

Proof. We prove the theorem by showing that a special case of the problem is NP-complete. Suppose that there are n^2 VM requests and there is one VM request at each time-slot, which means that $K(\tau) = 1$ and $M(\tau) = \tau$. So, the energy consumption can be described as:

$$E = \frac{a * \Delta t}{\mathcal{C}} \sum_{\tau=1}^{n^2} \sum_{v=1}^{\tau} R_v + b * \Delta t \sum_{\tau=1}^{n^2} N(\tau).$$

Let the resource requirement of jth$(n \leq j \leq n^2)$ VM be equal to the capacity of PM, which means that $R_v^d = \mathcal{C} (1 \leq d \leq D)$. So, we have $N(\tau + 1) = N(\tau) + 1 (n \leq \tau < n^2)$. The energy consumption should be:

$$E = \frac{a}{2} * \Delta t * n^2 * (n^2 + 1) + b * \Delta t \left(\sum_{\tau=1}^{n} N(\tau) + (n-1) * n * N(n) + \sum_{i=1}^{n(n-1)} i \right).$$

Obviously, we have $N(\tau) \leq N(n)$ $(1 \leq \tau \leq n)$. From the above equation, we know that to minimize the value of E, it is necessary to minimize the value of $\sum_{\tau=1}^{n} N(\tau) + (n-1)nN(n)$. Next, we prove that we must minimize $N(n)$ to get the minimal value of $\sum_{\tau=1}^{n} N(\tau) + (n-1)nN(n)$.

According the setting, we have $0 \leq N(\tau+1) - N(\tau) \leq 1$, and $N(\tau) < \tau$. Assuming h and k are two potential values of $N(n)$, without loss of generality, let $0 < h < k \leq n$. So,

$$\begin{aligned} f &= \left(\sum_{\tau=1}^{n} N_k(\tau) + (n-1) * n * k\right) - \left(\sum_{\tau=1}^{n} N_h(\tau) + (n-1) * n * h\right) \\ &= \sum_{\tau=1}^{n} N_k(\tau) - N_h(\tau) + n(n-1)(k-h). \end{aligned}$$

$N_k(\tau)$ and $N_h(\tau)$ are two potential sequences of $N(\tau)(1 \leq \tau \leq n)$, which satisfy the basic constraints: $0 \leq N(\tau+1) - N(\tau) \leq 1$, and $N(\tau) < \tau$. So, we have:

$$\sum_{\tau=1}^{n} N_k(\tau) - N_h(\tau) \geq \sum_{i=1}^{k-h} i + (n-k)(1-h) > n(1-h).$$

And easily, we have $n(n-1)(k-h) > n(n-1)$. Hence, we have:

$$f > (1-h) + n(n-1) = n(n-h) > 0.$$

So, it is easy to imply that, to get the minimal E, we must get the minimized $N(n)$. The minimized $N(n)$ is the optimal solution of the offline virtual machine placement problem while given the first n VM requests, which is a typical offline vector bin packing problem, and the problem has been proved to be NP-complete.

According to the special case above, we conclude that the online virtual machine placement problem is NP-complete. □

As we learned from Eq. (2), we know that minimizing $N(\tau)$ can significantly reduce energy consumption. Though it is hard to get an optimal solution, we can easily give a lower bound for the placement.

$$N(\tau) \geq \max_{1 \leq d \leq D} \left\{ \frac{1}{\mathcal{C}} * \sum_{v=1}^{M(\tau)} R_v^d \right\}. \tag{3}$$

4. Multi-dimensional space partition model

PM consists of D-dimensional resources; it will be fully loaded if an arbitrary dimensional resource is exhausted. Then, new PM starts up to host other VMs.

There may be resource fragments in the fully loaded PM. Obviously, the resource fragment is wasted, and the phenomenon of resource wasting is named a *resource leak* in this paper. In order to minimize the number of PMs under a given VM sequence, it is necessary to reduce the *resource fragment*.

To represent the D-dimensional resource utilization of PM, we propose a multi-dimensional space partition model, which can also judge the suitability of resource utilization for the VM placement, and thereby guide VM placement. The basic idea of a multi-dimensional space partition model is that it characterizes a *resource leak* quantitatively. At first, we formally define the *usage state* of PM.

Definition 2 (*Usage State*)**.** Given a PM p, the dth dimensional resource utilization ratio is $\gamma_d (1 \leq d \leq D)$. The *usage state* of PM can be represented as $us(p) = (\gamma_1, \gamma_2, \ldots, \gamma_D)$.

Each *usage state* corresponds to one *point* in the multi-dimensional space partition model; the *point* can also be presented as $(\gamma_1, \gamma_2, \ldots, \gamma_D)$. The notions of *usage state* and *point* are interchangeable in this paper.

Fig. 1 shows an example when $D = 2$. The point $\boldsymbol{E}$ with coordinates (1.0, 1.0) refers to the *usage state* that all dimensional resources are exhausted, while the point **O** is the initial point, which means that the PM is idle. There are three domains in the partition model.

Acceptance Domain (AD): The acceptance domain implies that the D-dimensional resources are all nearly exhausted, and that there are few resource fragments in the PM. It is an ideal case for all PMs, and the PMs whose corresponding posterior usage state lies in this domain have a higher priority for being selected.

Forbidden Domain (FD): The forbidden domain indicates that there is an obvious disequilibrium of D-dimensional resource utilization, and there may be excessive resource fragments in the PM. This case should be avoided.

Safety Domain (SD): The safety domain means that there is no obvious disequilibrium of D-dimensional resource utilization. This is the balanced case.

The unit square is partitioned into three parts by three quarter circles. The acceptance domain part is one quarter circle with center $\boldsymbol{E}$ and radius $r_0 (r_0 \in [0, 1.0])$. The forbidden domain part is divided into two subparts by two symmetrical

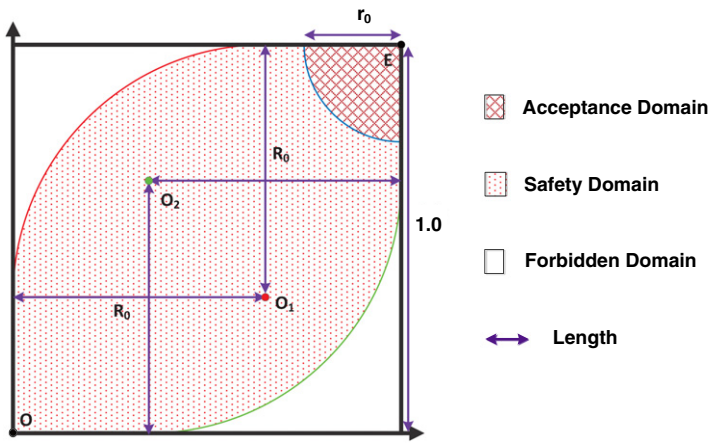


Fig. 1. Multi-dimensional space partition model.

quarter circles, with the same radius $R_0 (R_0 \in [0, 1.0])$. $\mathbf{O_1}$ (the red point) is the center of the red quarter circle and $\mathbf{O_2}$ (the green point) is the center of the green quarter circle. The safety domain lies in the space between the three quarter circles and square edges. The acceptance domain has a higher priority if there exists an overlapped zone.

Generally, for the D-dimensional space partition model, given point $\boldsymbol{S}(\gamma_1, \gamma_2, \ldots, \gamma_D)$ and the parameters r_0 and R_0, the domain determination function $\boldsymbol{f}$ can be defined as:

$$\boldsymbol{f}(S) = \begin{cases} AD, & \text{if } distance(S, E) \leq r_0; \\ SD, & \text{else if } (\forall d, distance(S, O_d) \leq R_0) \parallel \\ & (\forall d, \gamma_d \leq (1 - R_0)) \parallel \\ & (\forall d, \gamma_d \geq R_0); \\ FD, & \text{otherwise} \end{cases}$$

where $distance(\boldsymbol{S}, \boldsymbol{E})$ means the distance between the two points $\boldsymbol{S}$ and $\boldsymbol{E}(1, 1, \ldots, 1)$, so does $distance(\boldsymbol{S}, \boldsymbol{O_d})$, the coordinate of point $\boldsymbol{O_d}$ is represented as $(\alpha_1, \alpha_2, \ldots, \alpha_D)$, where

$$\alpha_k = \begin{cases} 1 - R_0, & \text{if } k = d; \\ R_0, & \text{otherwise.} \end{cases}$$

5. An energy efficient virtual machine placement algorithm with balanced resource utilization

In this section, we propose an online algorithm, EAGLE, based on the partition model. The basic idea of EAGLE is to make a tradeoff between balancing multi-dimensional resource utilization and minimizing the *local* number of PMs when placing VMs at each time-slot. The basic operation is starting up a new PM to avoid excessive *resource fragments*. It is efficient in decreasing resource fragments over the long run, and further reduces the number of PMs. Algorithm 1 shows the framework of EAGLE.

5.1. PM selection

At the τth time-slot, the data center accepts $K(\tau)$ VM requests. Some PM(s) will be selected to host the VM(s). According to the *usage state* of PM and VM *resource volume*, we define the posterior usage state as follows:

Definition 3 (*Posterior Usage State*)**.** Given a PM p, which has sufficient resources to host a VM v, the D-dimensional resource of VM v is $(R_1, R_2, \ldots, R_D)$. The usage state of PM is $(\gamma_1, \gamma_2, \ldots, \gamma_D)$ and the capacity is $\mathcal{C}$. The posterior usage state refers to the new usage state when the VM is placed on the PM, and the posterior usage state can be represented as $pus(p, v) = (\gamma'_1, \gamma'_2, \ldots, \gamma'_D)$, where $\gamma'_d = (\mathcal{C} * \gamma_d + R_d)/\mathcal{C}$.

Posterior usage state indicates the suitability of the PM for hosting the VM. The suitability is determined by the domain in which the *posterior usage state* lies in the partition model. The PM, whose posterior usage state lies in the *acceptance domain (AD)*, has the priority to be selected, while that which lies in the *safety domain (SD)* has secondary priority. The posterior usage state laying in *forbidden domain (FD)* implies that an excessive *resource leak* will occur in the PM, and the PM will not be selected. A new PM will start up if there is no PM is selected.

Algorithm 1 Framework of EAGLE

initial running PM number N, initial time-slot τ
fetch the VM requests $R(\tau)$
while $R(\tau)! = NULL$ **do**
  $K(\tau) \leftarrow R(\tau).size$
  for $k = 1$ to $K(\tau)$ **do**
    if $\forall p \exists p_0$, $pus(p, k) \in AD \wedge \mathfrak{R}(p_0) = \max\{\mathfrak{R}(p)\}$ **then**
      place the VM on the P_0^{th} PM
    else if $\forall p \exists p_1$, $pus(p, k) \in SD \wedge \mathfrak{D}(p_1) = \min\{\mathfrak{D}(p)\}$ **then**
      place the VM on the P_1^{th} PM
    else
      start new PM and place VM on it
      $N \leftarrow N + 1$
    end if
  end for
  record $N(\tau)$
  $\tau \leftarrow \tau + 1$
  fetch $R(\tau)$
end while

5.2. Metric determination

There may be many PMs with the same priority; the one with optimal suitability will be selected. We provide the definitions of the *metric* $\mathfrak{R}$ and $\mathfrak{D}$, as follows:

Definition 4 ($\mathfrak{R}$*: Resource Utilization Ratio*)**.** Given PM with *(posterior) usage state* $(\gamma_1, \gamma_2, \ldots, \gamma_D)$, $\mathfrak{R}$ is defined as:

$$\mathfrak{R} = \frac{1}{D} \sum_{d=1}^{D} \gamma_d.$$

The *metric* $\mathfrak{R}$ represents the mean value of the D-dimensional resource utilization ratios, and it indicates the overall resource utilization of the PM.

Definition 5 ($\mathfrak{D}$*: Distance*)**.** Given PM with *(posterior) usage state* $(\gamma_1, \gamma_2, \ldots, \gamma_D)$, $\mathfrak{D}$ is defined as:

$$\mathfrak{D} = \sqrt{\sum_{d=1}^{D} \left(\gamma_d - \frac{1}{D} \sum_{d=1}^{D} \gamma_d \right)^2}.$$

The *metric* $\mathfrak{D}$ represents the distance from the *usage state* point to the line $x_1 = x_2 = \cdots = x_D$, and it indicates the equilibrium of D-dimensional resource utilization.

According to the *metrics*, some PM is selected, or a new PM starts up. It is unavoidable for PMs that there exists *resource fragment* during the online placement process. However, excessive *resource leak* can be avoided by balancing the D-dimensional resource utilization.

5.3. Comments

According to the basic idea of EAGLE, we know that there is a tradeoff between balancing multi-dimensional resource utilization and minimizing the local number of PMs. The multi-dimensional space partition model is affected by two parameters R_0 and r_0. We name R_0 *balance factor*, and name r_0 *satisfaction factor*. *Balance factor* indicates the expectation of the degree of equilibrium, and quantitatively defines the equilibrium. *Satisfaction factor* indicates the expectation of the degree of local optimization, and determines the area of *acceptance domain*.

6. Performance evaluation

In this section, we first give the power model of PM, according to sufficient experiments. To compare with the proposed balanced algorithm, we implement another algorithm with greedy manner, the first fit algorithm. First fit is a well known algorithm for the bin packing problem, and it has been proven to show satisfactory performance. We present the comparative

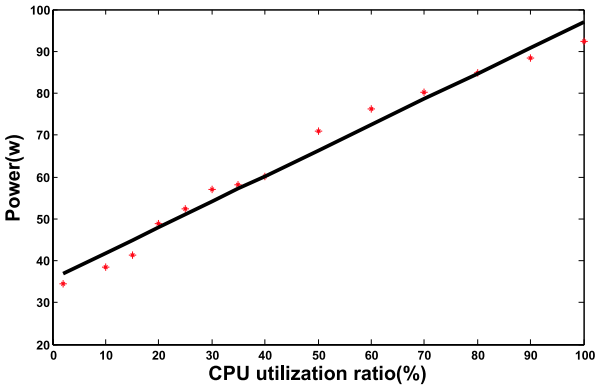


Fig. 2. Relationship between power and CPU utilization ratio.

results between the balanced algorithm EAGLE and first fit algorithm. Also, we investigate the impact of *balance factor* R_0 and *satisfaction factor* r_0. With extensive simulations and real trace experiments, we conclude that the idea of balancing resource utilization is effective for energy saving.

6.1. Power model

The purpose of this experiment is to establish a realistic power model to provide an explicit relationship between the power consumption and the amount of resources used.

We measure the power of PM via a power monitor device, and gather a lot of real data including power, CPU utilization ratio, and memory occupation. We analyze the data and establish a model of power consumption. We found that power is mainly determined by the utilization of CPU; Fig. 2 shows an sample of the data. According to plentiful analysis and verification, we confirm that the conclusion is:

$$P = a * r + b$$

where r is *CPU utilization ratio* and $r \in (0, 1]$, and a and b are constants associated with the special resource configuration of PM. For example, for the sample in Fig. 2, the values are: $a = 61.24$, $b = 35.70$.

6.2. First fit algorithm

While placing the VMs, first fit algorithm can be seen as a locally optimal algorithm in greedy manner. The framework of the implemented online algorithm is shown in Algorithm 2. When a VM request arrives, the firstly scanned one with sufficient available resources will be selected to host the VM. To avoid unnecessary scanning, the fully loaded PMs are not scanned. If there is no PM with sufficient available resources, a new PM will start up and create the requested VM on the newly-started PM. It is a greedy manner to place the VMs and, by this way, we can achieve a local optimization.

6.3. Simulation setup

In our simulations, we set two kinds of scenarios, single VM request and multiple VM requests, at each time-slot. The details of the scenarios are described as follows.

Single VM request. There is one VM request at each time-slot. We make $D = 3$ while adopting the D-dimensional space partition model in the simulations, and the three types of resources are independent of each other. We take two data sets into account.

For the first data set, we let the resource requirement follows the uniform distribution, which is also adopted in [14]. The resource requirement of each dimension varies from 20 to 80, and the capacity of each dimensional resource is 150. We illustrate the results of 10,000 VM requests.

In the second data set, we consider the resource requirement to be equal to one of the VM instances provided by Amazon EC2 [30]. We take 9 instance types into account in the simulation, including 4 standard instances, 3 high-memory instances, and 2 high-CPU instances. The server capacity is set as that the server can host 9 different VMs, one for each VM instance type.

For each of the data sets, there are many instances with the same distribution and parameter settings. The experimental result refers to the average value of each data set.

Multiple VM requests. There are $K(\tau)$ VM requests in the τth time-slot, and $K(\tau)$ is a random value between 60 and 100. We use the D-dimensional space partition model in our algorithm, and make $D = 2$; it can be considered as *CPU* and *memory*. The capacity of the resource is 12, it is consistent with the real situation that a physical server owns 12 GB (or 24 GB) memory

Algorithm 2 Framework of Online First Fit Algorithm

```
initial running PM number N, initial time-slot τ
fetch the VM requests R(τ)
while R(τ)! = NULL do
  K(τ) ← R(τ).size
  for k = 1 to K(τ) do
    for n = 1 to N(τ) do
      if the n^th PM has sufficient resources to host the k^th VM then
        place the VM on the n^th PM
        break;
      end if
    end for
    start new PM and place VM on it
    N ← N + 1
  end for
  record N(τ)
  τ ← τ + 1
  fetch R(τ)
end while
```

and 12 (or 24) CPU cores. The required resource is a random value between 1 and 8. It is reasonable that different tasks have variant resource requirements, e.g., some computational tasks may be sensitive to CPU, while some web services need more memory. Precisely because of this differentiation, the disequilibrium of multi-dimensional resource utilization exists, and then it results in a *resource leak*.

Based on the different scenarios, we look into the two concerns: (1) How the number of the PMs varies when the VM requests reach the data center continuously under different settings? (2) How the multi-dimensional space partition model affects the placement procedure and how the model parameters, balance factor, and satisfaction factor affect the model efficiency?

6.4. Simulation results: single VM request

We implemented the algorithms under the scenario setting mentioned above. For the single VM request case, Fig. 3 shows the results of the first data set. The horizontal axis of each sub-figure is the number of VM requests, which can also be regarded as time-slots, because there is just one VM request at each time-slot. We use the metric $N(\tau)$, the number of running PMs at the τth time-slot, to evaluate the performance of the algorithms; Fig. 3(a) shows the result of $N(\tau)$. Fig. 3(b) exhibits the results measured by power consumption. Both the balanced algorithm EAGLE (black dashed line) and greedy algorithm first fit (green line) have a similar growth pattern as the number of VM requests increases. However, the results of EAGLE rise slower than the first fit algorithm.

Fig. 4 shows the simulation results when the requested VM is one of the 9 VM instances from Amazon EC2. Similar to Fig. 3, the result is illustrated with the same metric $N(\tau)$. We evaluate the performance of our algorithm via 50,000 VM requests, with one for each time-slot.

Let us inspect the gap between the two curves. We learn from the figure and our experimental data, that the balanced algorithm EAGLE can save nearly 10% as much energy, compared to the first fit algorithm. This is a significant saving when considering the scale of the big data center.

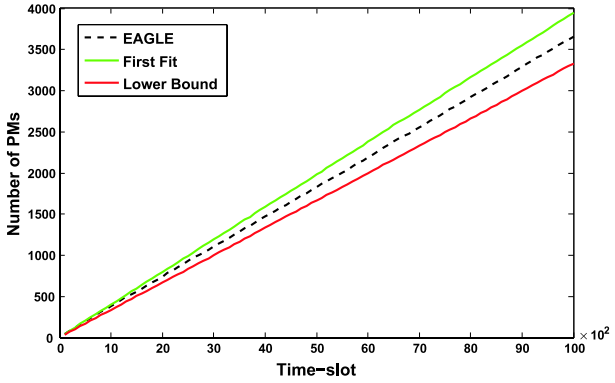
The uniform distribution of resource requests makes the results smooth; we will give more detailed comparative results and discussions in the next subsection.

6.5. Simulation results and analysis: multiple VM requests

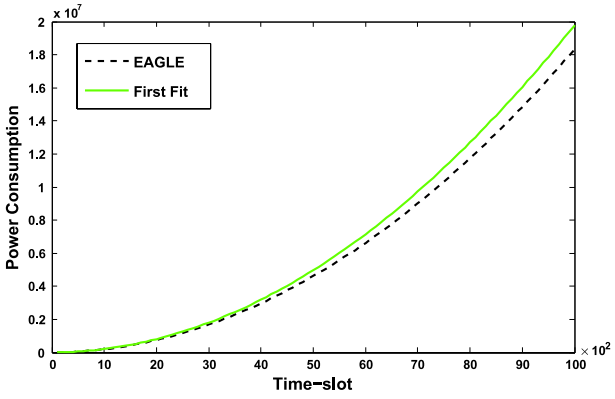
We deal with the case of multiple VM requests and give more detailed comparative results. Besides the first fit algorithm, we also give the value of lower bound, as mentioned in Eq. (3), to evaluate the performance of balanced algorithm EAGLE.

We use the metric $N(\tau)$, the number of running PMs at the τth time-slot, to evaluate the performance of the algorithms. The simulation results are shown in Figs. 5 and 6, the x-coordinate of the figures refers to the time-slot which starts from 0 and ends when all of the VMs have been hosted. As we know from Fig. 5(a), which shows the result of $N(\tau)$, the performance of EAGLE (black line) is very close to the *lower bound* (cyan line). As time-slot increases, there is a visible gap between balanced algorithm EAGLE and first fit algorithm (green line).

To illustrate the performance of EAGLE directly, Fig. 5(b) shows the comparative results: the ordinate axis denotes the ratio of $N(\tau)$ associated with *first fit* algorithms to the $N(\tau)$ determined by *EAGLE*. At the primary stage of placement, the performance fluctuates. Then, the lines flatten out after the period of fluctuation. Compared to *EAGLE, first fit* algorithm



(a) Number of PMs.



(b) Power consumption.

Fig. 3. Simulation results under the settings: sample size is 10,000, min. = 20, max. = 80.

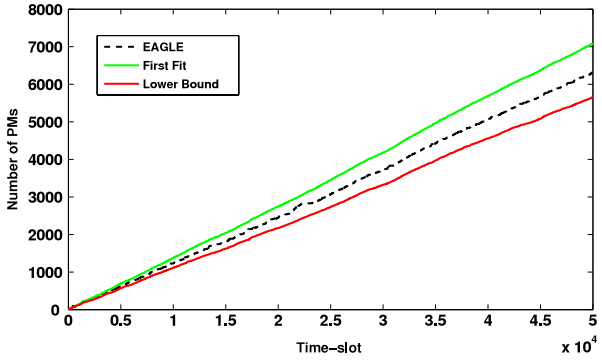


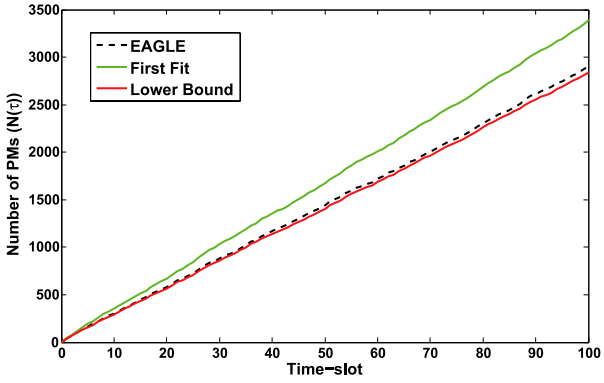
Fig. 4. Simulation result when the resource requirement is extracted from the VM instances of Amazon EC2.

needs 1.15 times PMs. This indicates that we can save a near 15% energy cost via adopting the balanced algorithm EAGLE to accomplish the placement task.

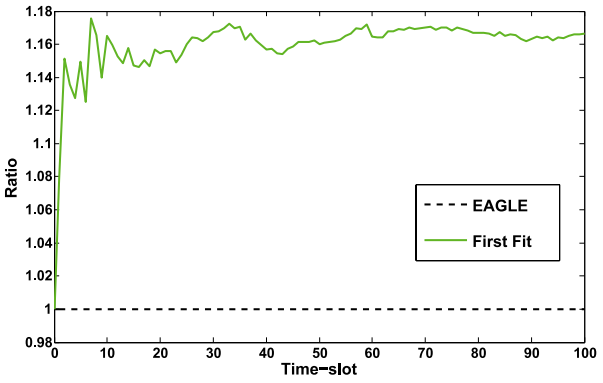
The energy saving benefits from the avoidance of resource leak. Fig. 6 shows the results of *resource fragment* for each algorithm. The ordinate axis represents the accumulated value of resource fragments distributed in all of the $N(\tau)$ PMs. It is clear that there exists more *resource fragments* while adopting *first fit* algorithm.

It is obvious that these shown results (Figs. 5 and 6) are the good ones among the results corresponding to different parameters. In fact, we also analyze the impact of model parameters (balance factor and satisfaction factor) in our experiments.

Impact of balance factor R_0. The comparative results of various R_0 with fixed r_0 are shown in Fig. 7. Fig. 7(a) shows the case when $r_0 = 0.1$, while the case when $r_0 = 0.0$ is shown in Fig. 7(b). The case when $r_0 = 0.1$, $R_0 = 1.0$ is adopted in the related simulations of Fig. 5. The ordinary axis is the ratio of the corresponding $N(\tau)$ of various R_0 to the $N(\tau)$ associated



(a) $N(\tau)$: number of PMs.



(b) Ratio of $N(\tau)$: first fit to EAGLE.

Fig. 5. Performance comparison of different algorithms.

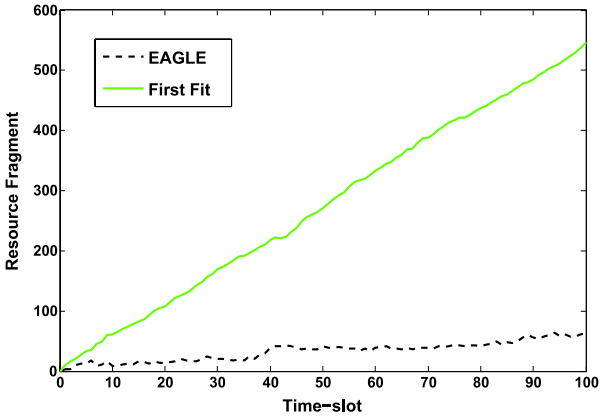
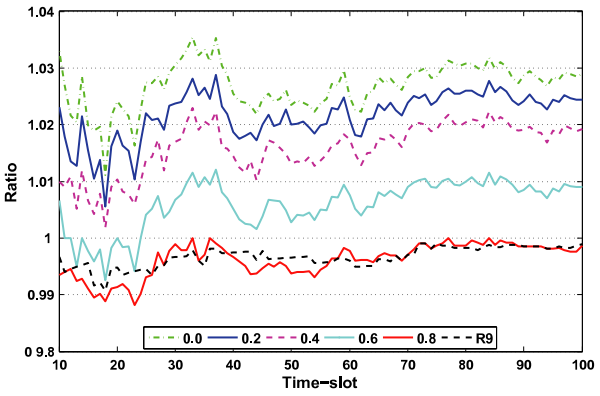
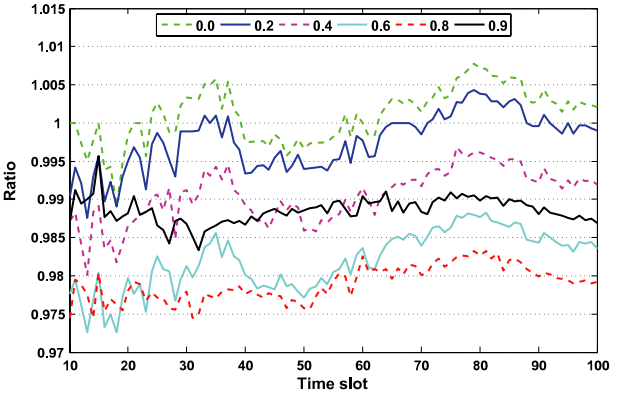


Fig. 6. Resource fragment: EAGLE vs. FF.



(a) $r_0 = 0.1$.



(b) $r_0 = 0.0$.

Fig. 7. Impact of *balance factor* R_0. r_0 is fixed, R_0 varies.

with $R_0 = 1.0$. Learning from the results, both $r_0 = 0.0$ and $r_0 = 0.1$, the performance improves when the value of R_0 varies from 0.0 to 0.8. The result indicates that it is better to enhance the balance restriction, i.e. increase *balance factor* R_0; we can benefit from balancing resource utilization. However, if the balance restriction is immoderate, e.g. the case when $R_0 = 1.0$, there will be a negative effect.

Impact of satisfaction factor r_0. Fig. 8 shows the comparative results of various r_0 when $R_0 = 0.8$, the best choice according to the results in Fig. 7. The ordinary axis is the ratio of the corresponding $N(\tau)$ of various r_0 to the $N(\tau)$ associated with $r_0 = 0.1$. The performance improves when the value of r_0 varies from 0.9 to 0.1. The result indicates that it will be better if

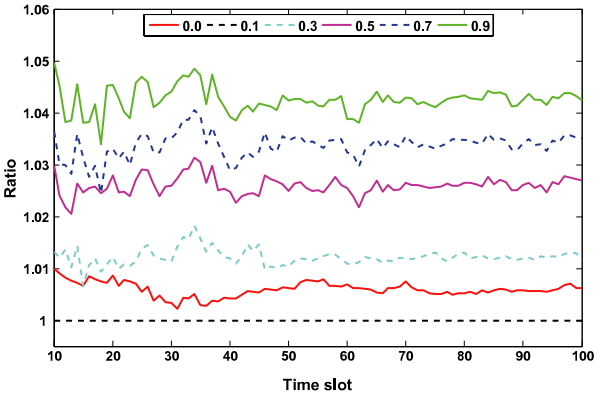
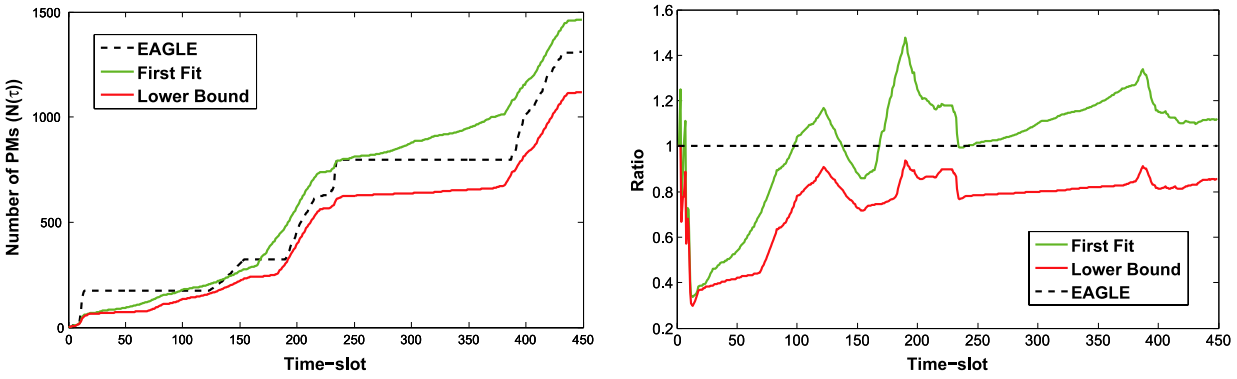


Fig. 8. Impact of *satisfaction factor* r_0. R_0 is fixed, r_0 varies.



(a) $N(\tau)$: number of PMs.

(b) Ratio of $N(\tau)$: first fit (lower bound) to EAGLE.

Fig. 9. Google cluster data.

we weaken the local optimization, i.e. decrease *satisfaction factor* r_0. However, if we ignore the local optimization completely, e.g. the case when $r_0 = 0.0$, there will be some negative effect.

The simulation results of Figs. 7 and 8 demonstrate that there indeed exists a tradeoff between balancing resource utilization and local optimization.

6.6. Case study: real resource trace

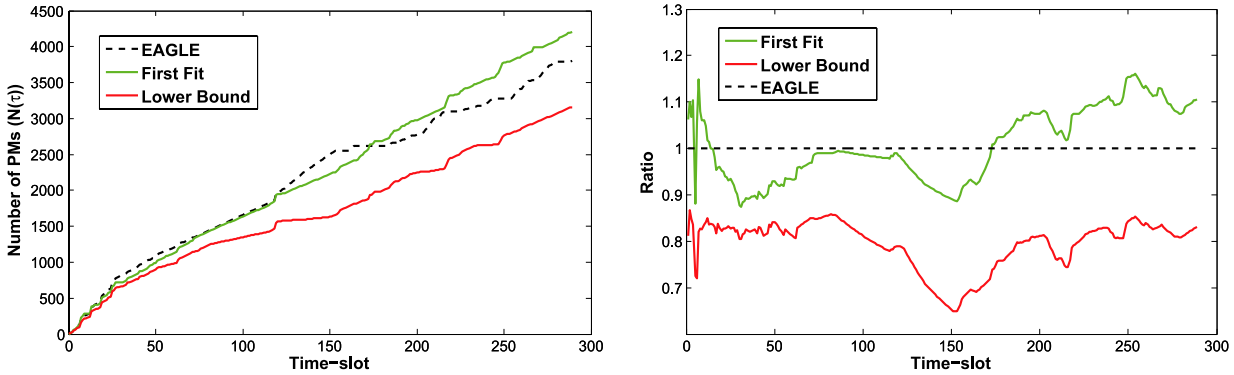
To evaluate the performance of EAGLE, we also run our experiments using two widely-used real traces from Google [31] and another large computing cluster [32].

The Google dataset provides traces over a 7-h period, and we extract 45,000 continuous items from the dataset. Each item corresponded to a task resource requirement, which contains normalized task cores and normalized task memory. We consider the normalized task cores and normalized task memory as two dimensions of the model, shown in Section 4.

Fig. 9 shows the results of the Google dataset. The three curves in Fig. 9(a) represent the values of $N(\tau)$ given by the EAGLE algorithm (black line), first fit algorithm (green line), and lower bound (red line). There is an obvious gap between the black line (EAGLE) and the green line (first fit) after the 250th time-slot. We can see that there are three flat segments, which means that the number of running PMs is not increasing as time-slot goes. That is because enough PMs have been started up before the flat point, to avoid imbalanced usage of multi-dimensional resources; here this means CPU and memory. The appearance of these flat segments indicates that there exist many imbalanced resource requirements, while EAGLE makes a tradeoff between local optimization and long-run optimization; we indeed can benefit long-run optimization from balanced resource utilization. We also exhibit the comparative results, the ratio of $N(\tau)$ corresponding to first fit algorithm (lower bound) to EAGLE, as shown in Fig. 9(b). At the beginning of the placement process, first fit needs less PMs than EAGLE, which is nearly the value of the lower bound. However, as more imbalanced resource requirements arrive, the number of PMs increase dramatically.

Besides the Google cluster data, we also experiment on another real trace dataset of real parallel workloads from production systems [32].

The other large computing cluster dataset we use contains logs of real parallel workloads from production systems. We extract 30,000 items from the newest RICC log (RICC dataset). This is similar to the Google dataset, for each item contains



(a) The value of $N(\tau)$.

(b) Ratio of $N(\tau)$: first fit (lower bound) to EAGLE.

Fig. 10. Real parallel workloads from production systems.

a resource requirement (number of processors and memory), but the resource requirements in the RICC dataset are not normalized. We set the PM capacity as its physical server mentioned in the source.

The results of the RICC dataset are shown in Fig. 10. We can find a similar fact, the EAGLE algorithm performs well after some time-slot between 150 and 200, though there are not as many imbalanced resource requirements in this workload trace.

By comparing the results of the real trace (Figs. 9 and 10) with the simulation results (Figs. 3–5), we believe that the simulation matches the real trace well, and that the idea of balancing resource utilization indeed works.

7. Conclusion

In this paper, we address the problem of online virtual machine placement with the goal of minimizing the total energy consumption. Our basic aim is to reduce the number of resource fragments and decrease their sizes, as well. So we present the multi-dimensional space partition model to describe the resource usage state of PMs. Based on the model, we further propose an energy efficient VMP algorithm EAGLE, which can balance the utilization of multi-dimensional resources, lower down the number of running PMs, and thus cut down energy consumption. The effectiveness and efficiency of EAGLE are validated via extensive experiments.

Acknowledgments

This work is supported in part by the National Natural Foundation of China under Grant No. 61073028, 61021062, 61202113; the National Basic Research Program of China (973) under Grant No. 2009CB320705; the Key Technology Research and Development Program of Jiangsu under Grant No. BE2010179; Jiangsu Natural Science Foundation under Grant No. BK2011510; the Fundamental Research Funds for the Central Universities under Grant No. 1127020229; US NSF grants ECCS 1128209, CNS 1138963, CNS 1065444, and CCF 1028167.

References

[1] L. Ogiela, M.R. Ogiela, Advances in Cognitive Information Systems, in: Cognitive Systems Monographs, vol. 17, Springer-Verlag, Berlin, Heidelberg, 2012.
[2] A.M. Meystel, J.S. Albus, Engineering of Mind: An Introduction to the Science of Intelligent Systems, John Wiley & Sons, 2001.
[3] J. Kim, K. Kim, J.H. Park, T. Shon, A scalable and privacy-preserving child-care and safety service in a ubiquitous computing environment, Mathematical and Computer Modelling 55 (1–2) (2012) 45–57.
[4] D.G. Zhang, Y.P. Liang, A kind of novel method of service-aware computing for uncertain mobile applications, Mathematical and Computer Modelling 57 (3–4) (2013) 344–356.
[5] A. Greenberg, J. Hamilton, D.A. Maltz, P. Patel, The cost of a cloud: research problems in data center networks, ACM SIGCOMM Computer Communication Review 39 (1) (2009) 68–73.
[6] X. Bai, K. Wang, C. Ding, J. Li, Context-aware integration of data-centerd services, in: Porc. IEEE International Conference on Service-Oriented Computing and Applications, SOCA, 2009.
[7] L. Zheng, M. Li, C. Wu, H. Ye, R. Ji, X. Deng, Y. Che, C. Fu, W. Guo, Development of a smart mobile farming service system, Mathematical and Computer Modelling 54 (2011) 1194–1203.
[8] S.C. Misra, A. Mondal, Identification of a company's suitability for the adoption of cloud computing and modelling its corresponding return on investment, Mathematical and Computer Modelling 53 (4–5) (2011) 504–521.
[9] D. Kusic, J. Kephart, J. Hanson, N. Kandasamy, G. Jiang, Power and performance management of virtualized computing environments via lookahead control, Cluster Computing 12 (1) (2009) 1–15.
[10] J.T. Piao, J. Yan, A network-aware virtual machine placement and migration approach in cloud computing, in: Proc. of 9th International Conference on Grid and Cloud Computing, GCC, 2010.
[11] E. Bin, O. Biran, O. Boni, E. Hadad, E.K. Kolodner, Y. Moatti, D.H. Lorenz, Guaranteeing high availability goals for virtual machine placement, in: Proc. the 31st International Conference on Distributed Computing Systems, ICDCS, 2011.

[12] M. Wang, X. Meng, L. Zhang, Consolidating virtual machines with dynamic bandwidth demand in data center, in: Proc. the 30th Conference on Computer Communications, INFOCOM, 2011.
[13] X. Meng, V. Pappas, L. Zhang, Improving the scalability of date center networks with traffic-aware virtual machine placement, in: Proc. the 29th Conference on Computer Communications, INFOCOM, 2010.
[14] J.W. Jiang, T. Lan, S. Ha, M. Chen, M. Chiang, Joint VM placement and routing for data center traffic engineering, in: Proc. the 31st Conference on Computer Communications, INFOCOM, 2012.
[15] U. Sharma, P. Shenoy, S. Sahu, A. Shaikh, A cost-aware elasticity provisioning system for the cloud, in: Proc. the 31st International Conference on Distributed Computing Systems, ICDCS, 2011.
[16] R. Bianchini, R. Rajamony, Power and energy management for server systems, IEEE Computer 37 (11) (2004) 68–74.
[17] W. Vogels, Beyond server consolidation, ACM Queue 6 (1) (2008) 20–26.
[18] C. Hyser, B. McKee, R. Gardner, B.J. Watson, Autonomic virtual machine placement in the data center, HP Labs Technical Report, February 2008.
[19] A.V. Do, J. Chen, C. Wang, Y.C. Lee, A.Y. Zomaya, B.B. Zhou, Profiling applications for virtual machine placement in clouds, in: Proc. the 2nd International Conference on Cloud Computing, GRIDs, and Virtualization, 2011.
[20] G.J. Woeginger, There is no asymptotic PTAS for two-dimensional vector packing, Information Processing Letters 64 (6) (1997) 293–297.
[21] J. Csirik, J.B. Frenk, M. Labbe, S. Zhang, On the multidimensional vector bin packing, Acta Cybernetica 9 (4) (1990) 361–369.
[22] C. Chekuri, S. Khanna, On multi-dimensional packing problems, in: Proc. 10th Anual ACM–SIAM Symposium on Discrete Algorithms, SODA, 1999.
[23] W.F. de la Vega, G.S. Lueker, Bin packing can be solved with $1 + \epsilon$ in linear time, Combinatorica 1 (4) (1981) 349–355.
[24] D. Karger, K. Onak, Polynomial approximation schemas for smoothed and random instances of multi-dimensional packing problem, in: Proc. 18th Anual ACM–SIAM Symposium on Discrete Algorithms, SODA, 2007.
[25] S.S. Seiden, On the online bin packing problem, Journal of the ACM 49 (5) (2002) 640–671.
[26] A.C. Yao, New algorithms for bin packing, Journal of the ACM 27 (2) (1980).
[27] J. Zhu, D. Li, J. Wu, H. Liu, Y. Zhang, J. Zhang, Towards bandwidth guarantee in multiple-tenancy cloud computing networks, in: Proc. the 20th IEEE International Conference on Network Protocols, ICNP, 2012.
[28] M. Alicherry, T.V. Lakshman, Network aware resource allocation in distributed clouds, in: Proc. the 31st Conference on Computer Communications, INFOCOM, 2012.
[29] X. Li, Z. Qian, R. Chi, B. Zhang, S. Lu, Balancing resource utilization for continuous virtual machine requests in clouds, in: Proc. the 6th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, IMIS, 2012.
[30] Amazon EC2 instance types. http://aws.amazon.com/ec2/.
[31] Google cluster data. http://googleresearch.blogspot.com/2010/01/google-cluster-data.html.
[32] Logs of real parallel workloads from production systems. http://www.cs.huji.ac.il/labs/parallel/workload/logs.html.