entropy

MDPI

Article

A Cross-Entropy-Based Admission Control Optimization Approach for Heterogeneous Virtual Machine Placement in Public Clouds

Li Pan [1,*] and Datao Wang [2]

[1] School of Computer Science and Technology, Shandong University, Jinan 250101, China
[2] Jinan Resident Office of National Audit Office of the People's Republic of China, Jinan 250011, China; taowd_121@hotmail.com
* Correspondence: panli@sdu.edu.cn; Tel.: +86-137-9105-2549; Fax: +86-531-8839-0059

Academic Editor: Kevin H. Knuth
Received: 20 December 2015; Accepted: 10 March 2016; Published: 15 March 2016

Abstract: Virtualization technologies make it possible for cloud providers to consolidate multiple IaaS provisions into a single server in the form of virtual machines (VMs). Additionally, in order to fulfill the divergent service requirements from multiple users, a cloud provider needs to offer several types of VM instances, which are associated with varying configurations and performance, as well as different prices. In such a heterogeneous virtual machine placement process, one significant problem faced by a cloud provider is how to optimally accept and place multiple VM service requests into its cloud data centers to achieve revenue maximization. To address this issue, in this paper, we first formulate such a revenue maximization problem during VM admission control as a multiple-dimensional knapsack problem, which is known to be NP-hard to solve. Then, we propose to use a cross-entropy-based optimization approach to address this revenue maximization problem, by obtaining a near-optimal eligible set for the provider to accept into its data centers, from the waiting VM service requests in the system. Finally, through extensive experiments and measurements in a simulated environment with the settings of VM instance classes derived from real-world cloud systems, we show that our proposed cross-entropy-based admission control optimization algorithm is efficient and effective in maximizing cloud providers' revenue in a public cloud computing environment.

Keywords: cross-entropy; virtual machine; cloud computing; admission control

1. Introduction

Cloud computing delivers a new promising paradigm through which users can gain on-demand access to cloud services on a pay-per-use basis anytime and anyplace [1]. Cloud services have brought customers a dramatic shift in the way computing resources are used and allow for reduced operational and maintenance costs, shorter start-up time, increased reliability, and so on. Thus, more and more enterprises and end users are turning to outsource their jobs and applications into cloud data centers. Infrastructure as a Service (IaaS) is such a form of cloud computing service, by which cloud providers deliver computing resources over the Internet to consumers. On the other hand, cloud providers, who maintain pools of massive computing resources (e.g., computation, network, storage and software applications), can gain profits from users' payment through economies of scale [2].

In order to achieve efficient utilization of data center resources and improve cost-effectiveness, cloud providers usually need to consolidate multiple IaaS provisions into a single server. Virtualization technologies, such as VMWare [3] and OpenStack [4], make it possible to run multiple virtual machines (VMs) on a single physical machine (PM). By virtualization, every virtual machine appears to be a single and dedicated computer to a consumer, while in fact, it is not implemented as a single physical machine,

but uses part of the capacity of one or more physical servers. While fulfilling an IaaS request from a consumer, the cloud provider needs to create a VM by allocating resources, such as CPUs, memory, disk space, and so on, from physical servers, which is usually called *virtual machine placement*.

Since in an open and dynamic public cloud computing environment, cloud consumers usually have divergent requirements for leasing cloud computing resources, a cloud provider needs to offer several types of VM configurations, which are usually referred to as *instance types*. Different VM instance types have varying performance and, thus, should be charging different prices. Taking Amazon EC2 [5] as an example, it charges 0.026$ per hour of usage of a small VM instance type with one virtual CPU core and 1 GB of memory, while charging 0.18$ per hour of usage of a medium VM instance type with two virtual CPU cores and 4 GB of memory, *etc*. In such public cloud circumstances, serving the VMs of different instance types will bring varying revenues to a cloud provider. Thus, when there are a bunch of cloud consumers who request heterogeneous virtual machines, how to optimally place them into cloud data centers to maximize a cloud provider's revenue is a significant problem to solve. On the other hand, since the available physical resources operated by a cloud provider are usually fixed and, thus, limited, *admission control* can be employed as a general approach to prevent physical servers in a cloud data center from being overloaded.

In this paper, we propose a cross-entropy-based admission control approach to address the above heterogeneous virtual machines placement optimization problem. The proposed admission control mechanism can suggest to a provider which subset of VM service requests to accept and which to reject, when facing a set of VM requests with varying instance types. Briefly speaking, we formulate the VM admission control optimization problem as a multiple-dimensional knapsack problem, which is known to be NP-hard to solve. Thus, in this paper, we propose to use a cross-entropy-based optimization approach to obtain a feasible near-optimal solution to suggest to the cloud provider how many and what types of VMs to place into its data centers to achieve revenue maximization. Finally, we evaluate our cross-entropy-based VM admission control optimization mechanism in a simulated environment to illustrate the efficiency and effectiveness of the proposed approach.

The rest of this paper is organized as follows. In Section 2, we discuss the related work. The proposed IaaS-type cloud service provision platform and VM placement model with admission control are described in Section 3. Section 4 outlines the problem statement and the mathematical formulations for the heterogeneous VM admission control problem, while the cross-entropy-based optimization algorithm for solving the given revenue maximization problem is described in Section 5. The experimental evaluation is given in Section 6. Section 7 contains conclusions and future directions for this work, followed by acknowledgments and a list of references.

2. Related Work

The problem of placement, migration and scheduling of VMs across a cluster of PMs has been investigated in the past few years, and a number of related approaches have been proposed [6–8]. Bazarbayev *et al.* [9] propose a content-based scheduling algorithm for VM placement in cloud data centers, by utilizing similarity between VM disk images. Their approach can lower the network traffic during the transfers of VMs between racks in data centers and results in significant savings in data center network utilization and congestion. Luo *et al.* [10] analyze the network influence on distributed computing tasks and web applications and propose a self-adaptive network-aware virtual machine re-scheduling algorithm. Their proposed algorithm focuses on lowering communication cost among virtual machines through conditional and automatic virtual machine live migrations. Wang *et al.* [11] investigate the problem of cloud resource allocation and pricing and propose a suite of truthful auction-style pricing mechanisms. Addition, for VM allocation between multiple users, they propose a greedy resource allocation scheme to achieve reasonable economic and computational efficiency.

While there are a number of commercial IaaS cloud offerings on the market, they have not published their VM scheduling and allocation policies. Additionally, current open source cloud platforms and toolkits, such as OpenStack [4] and CloudSim [12], are generally equipped with

local-optimal, but simple VM allocation policies. For example, the CloudSim scheduler provides a first-come-first-sever (FCFS) policy for provisioning resources of physical machines to serve a VM creation request and selects the physical machine with the most available physical CPU cores as the best candidate for hosting the requested VM. While these VM allocation policies are efficient, they have not taken the economy factor into consideration, and thus, the revenue problem of cloud providers is not addressed by them. Our proposed revenue maximization heterogeneous VM placement approach can be implemented atop the schedulers on these open source cloud platforms and works as complementarities to those schedulers to allow the cloud providers to obtain higher profits from the physical resources in their cloud data centers.

Virtual machine migration has also received significant attention recently [13,14]. The work in [15] considers the inherent dependencies between VMs comprising a multi-tier application and introduces an efficient scheme for incorporating inter-VM dependencies and the underlying network topology into VM migration decisions. Bose *et al.* [16] propose to combine VM replication with scheduling to minimize VM migration latencies. They propose an algorithm that factors in de-duplication ratios amongst pairs of images while deciding on the replica placement of VM images, and they show that their approach can judiciously place the replicas of VM images at multiple cloud sites to minimize the storage requirements.

Applying the cross-entropy method for solving resource allocation problems in cloud computing environments is not completely new. Gaetano *et al.* [17] model cloud resource allocation as a stochastic optimization problem by leveraging the cross-entropy method and propose an approach based on cognitive heuristics to deal with risk minimizations and cost optimizations. They aim to address the problem of resource management in multi-purpose clouds, by reducing the cost associated with the execution of the users' applications. Compared to this work, the proposed approach in this paper aims at solving the heterogeneous virtual machine admission control problem to maximize the cloud provider's revenue. Additionally, the heterogeneity of prices and configurations in VM instances is a significant characteristic in current public cloud platforms, such as Amazon EC2 services [5], Aliyun [18], and so on.

As discussed above, research in the area of VM placement and scheduling in data center server farms focuses on different aspects of considerations, such as network transfer, communication cost and start-up latency during VM migrations. Our work is complementary to current VM placement approaches, and compared to these work, the primary contribution of our work lies in that we formulate the revenue maximization during VM admission control as a multiple-dimensional knapsack problem and propose to use a cross-entropy-based optimization approach to obtain a near-optimal eligible set. Additionally, the proposed approach in this work shows high efficiency and effectiveness for applying to real-world public cloud computing environments.

3. Framework

Figure 1 outlines the cloud IaaS platform architecture used in this paper. We here consider a cloud data center as a distributed cluster system built from a number of physical servers. These physical computing resources are pooled to serve multiple users' service requests. They can support the concurrent running of multiple and heterogeneous VM instances. An IaaS user requests a virtual infrastructure using a VM description template given by a VM instance type supported by the IaaS platform. A cloud provider can offer several VM instance types, each having an associated price for per-hour usage. The configuration of a VM instance type is defined in measurable terms, such as the number and speed of virtual CPU cores, the memory capacity, the size of storage space, the permissible bandwidths, and so on.

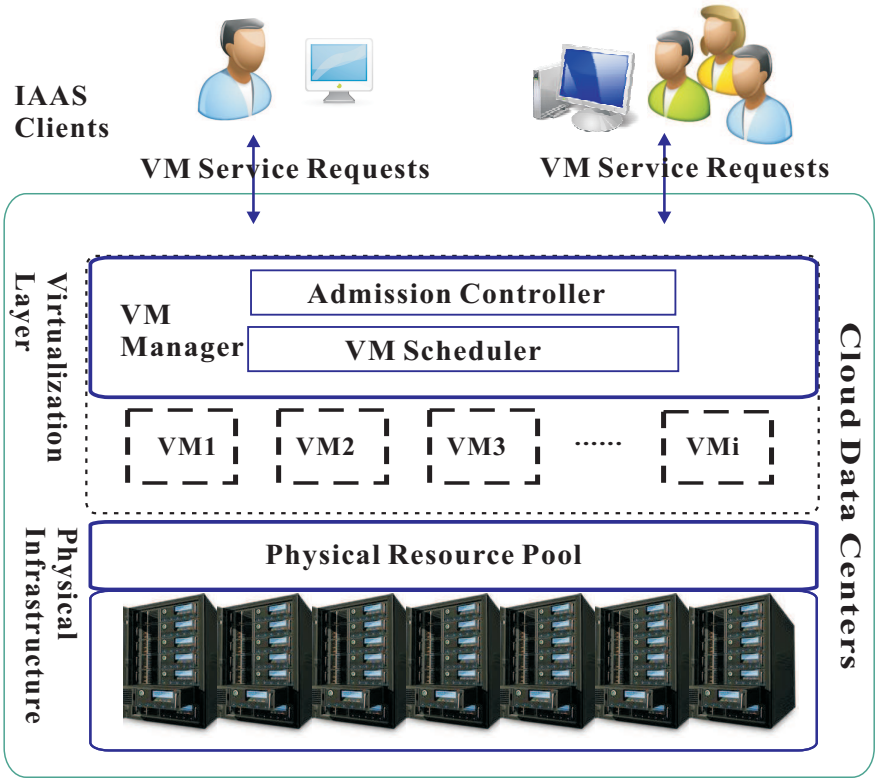

Figure 1. The architecture of the cloud IaaS platform.

The admission controller in a cloud data center employs an optimization engine equipped with our algorithm proposed in this paper to determine an eligible set from the waiting VM service requests in the system and maps them to a certain number of physical machines in a revenue-maximization manner. An IaaS cloud provider uses the admission control mechanism to decide whether to accept a client's VM service request or not. An admission controller can operate in rounds or once there are new VM service requests arriving at the cloud system. Additionally, in this paper, the admission controller works in rounds, *i.e.*, periodically with a specified time slot. In each round, the admission controller builds up an eligible set from all of the arriving VM service requests, which are concurrently submitted by multiple users, considering current VM instance types and pricing information, the current availability of physical resources in the cloud system, and so on. Such information can be collected by the virtual machine manager periodically from the cloud system.

The virtual machine manager schedules and assigns the VM service requests to physical servers in the cluster system, which is the so-called VM placement. It manages the process of VM instance creations with resources, such as CPU cores, memory and disk space allocated from physical servers, to serve the IaaS requests that have been accepted by the admission controller. In this paper, we assume each VM service request can be multiplexed and served by the physical infrastructure from the pools of multiple resources possessed by a collection of servers. If a client's VM service request is accepted by the provider, the client will make payment for per-hour usage of the allocated VM, according to the pricing information recorded in its associated VM instance specifications. The cloud provider aims to optimize the admission and placement of VMs into its cloud data centers to fulfill the submitted IaaS service requests amongst multiple users while maximizing its own revenue, which is obtained from the payments made by all users.

4. Formulation of the Problem

We introduce an optimal virtual machine admission control algorithm based on the VM placement problem described in the previous section. The goal of our admission control algorithm is to optimally

determine an eligible set of V IaaS requests with varying and heterogeneous VM instance types requested by a bunch of users, denoted as $V_1, V_2, ..., V_V$, across the available physical servers, within the resource constraints of physical machines, while maximizing the revenue gained from payments made by users for provisioning these VMs. The eligible set is selected from all of the waiting IaaS service requests submitted by multiple users. Without loss of generality and for simplicity, we assume that each user only submits one VM service request at a time slot, and we use n to denote the sum of users that are currently waiting to be served in the cloud system. We consider a fixed number of VM instance types delivered by a cloud provider, denoted as I. A VM instance type is characterized by m dimensions of virtual hardware resource configurations, denoted as $r_{j,1}, ..., r_{j,m}$. In this paper, for clarity and simplicity, we assume each virtual hardware dimension is additive. That is, the overall amount of the resource in a dimension required by all of the accepted VM service requests is the sum of resources each VM requires in that dimension. Additionally, each VM instance type it_k ($k \in I$) has an associated price for per-hour usage of a VM instance of this type, denoted as p_k. When a VM service request s_j is submitted by a user to the cloud provider, its required VM instance type is fixed, and we use ite_j to denote the index of its expected instance type. Thus, the required VM instance type demanded by service s_j can be represented as it_{ite_j}, and the amount of resources needed in dimension i to be allocated for hosting that VM can be represented as $r_{ite_j,i}$. Suppose c_i represents the overall amount of available resources in dimension i possessed by the physical servers in cloud data centers. When the total infrastructure capacity requirement exceeds the capacity of the available physical machines, the provider needs to decide which VM service requests to accept and which to deny.

As described above, our objective is to maximize the provider's revenue within its capacity constraints in determining an eligible set from the waiting VM service requests in the cloud system through admission control, and the maximization problem (**P1**) can be expressed as:

$$\textbf{maximize} \qquad \sum_{j=1}^{n} x_j p_{ite_j} \tag{1}$$

subject to:

$$\sum_{j=1}^{n} r_{ite_{j,i}} x_j \leq c_i, i = 1, ..., m \tag{2}$$

$$x_j \in \{0, 1\}, j = 1, ..., n \tag{3}$$

where x_j is a pseudo-Boolean integer decision variable to determine whether the VM service request from the j-th user is accepted or not. That is to say, x_j is one if the VM service request j is accepted and zero otherwise. The term p_{ite_j} in the objective function Equation (1) represents the price for the VM instance class required by user j. Equation (2) corresponds to the limits that the sum of the physical resources required by all of the accepted VM requests on every dimension cannot exceed the resource constraints on that dimension.

The above problem can be mapped to a multi-dimensional knapsack problem (MKP), which is one kind of NP-hard combinatorial optimization problem. For a small instance size of this kind of NP-hard combinatorial optimization problems, exhaustive search may be employed to find an optimal solution. However, for the above VM admission control problem, with more and more users turning to use IaaS cloud services, the number of VM service requests to be considered for admission is usually large, and thus, it would incur extra huge and even unacceptable computational cost for finding an optimal solution to the above revenue maximization problem. In this paper we propose a cross-entropy-based optimization algorithm to obtain a near-optimal solution, which we will discuss in detail in the next section.

5. Cross-Entropy-Based VM Admission Control Optimization Algorithm

In this section, the proposed cross-entropy-based VM admission control algorithm is illustrated. We first review how the basic cross-entropy (CE) method works and then present how we improve the original CE approach to solve the admission control optimization problem.

5.1. Review of Cross-Entropy Method

The CE method, which was pioneered by Reuven Rubinstein in 1997, is originated from the field of rare event simulation to estimate very small probabilities [19,20]. It is based on the concepts of cross-entropy and importance sampling. The basic CE method can be considered as an iterative procedure in which each iteration consists of two phases: (1) generate a random sample of data according to a pre-defined random mechanism; (2) and then based on the current data, update the parameters of the random mechanism to produce a new and better sample for the next iteration. During each iteration, the second phase involves minimizing the cross-entropy (or Kullback–Leibler (K-L)) distance.

Now, the CE methods are beginning to be applied for solving static and noisy combinatorial optimization problems. The basic idea for applying the CE method to the combinational optimization domain is considering the selection of an optimal solution from all of the feasible regions as a rare event and associating an estimation problem with the optimization problem. In this paper, we will not discuss all features of the CE method. Instead, we will only illustrate how the CE method is adapted to solve the VM admission control problem in the next subsection.

5.2. Cross-Entropy-Based Optimization Algorithm for MKP

In the previous section, we have shown that the optimization problem **P1** is NP-hard. We now develop a cross-entropy-based algorithm in order to find a near-optimal solution to the revenue maximization problem, following the basic idea of cross-entropy proposed in [19,20] as discussed above. The details of the cross-entropy-based VM admission control optimization algorithm (CEVMAC) are presented below.

The variable x_j $(j = 1, ..., n)$ in Equation (1) is a pseudo-Boolean integer decision variable with the values of zero or one. Thus, we can associate the above decision variables with the binary vectors $X = (X_1, ..., X_n)$, which are independent Bernoulli random variables with success probabilities $p_1, ..., p_n$. Here, p_j represents the probability that the j-th VM service request is admitted into the cloud data centers. Thus, we have $X \sim Ber(p)$, where $p = (p_1, ..., p_n)$, and it has a family of probability density function (pdfs) as:

$$f(X; p) = \prod_{j=1}^{n} p_j^{X_j} (1 - p_j)^{1 - X_j} \tag{4}$$

Now, in order to leverage the CE method to solve this optimization problem, we first need to cast it into the problem of the estimation of the probability of rare events. That is, consider the *associated stochastic problem* (ASP), by creating a sequence of parameter vectors $\widehat{p_1}, ..., \widehat{p_n}$ and then estimating the probability $\mathbb{P}_p(S(X)) > \gamma$ for a given level γ and the family of pdfs $\{f(\cdot; p)\}$, where $\mathbb{P}_p$ is the probability under which the random sample X has a probability density of $\{f(\cdot; p)\}$. Additionally, here, we use $S(X)$ to denote the objective function in the revenue maximization problem considered in this work. Then, for a given level γ, the goal is to find a reference parameter vector p so that the *Kullback–Leibler* distance between the two densities of $I_{S(X))\geq\gamma}$ and $f(\cdot; p)$, which is also termed the *cross-entropy* of these two densities, denoted as $D_{KL}(I_{S(X))\geq\gamma} \parallel f(\cdot; p))$, is minimized. Solving this cross-entropy minimization problem in the above revenue maximization problem with independent Bernoulli random variables is equivalent to solving the maximization problem:

$$\widehat{p^*} = \underset{v}{argmax} \frac{1}{N} \sum_{i=1}^{N} I_{S(X_i))\geq\gamma} \, lnf(X_i; p) \tag{5}$$

where X_i are generated from the probability density $f(\cdot; p)$. By combining Equation (4) with Equation (5), we can get:

$$\widehat{p_j^*} = \frac{\sum_{i=1}^{N} I_{S(X_i)\geq\gamma} X_{ij}}{\sum_{i=1}^{N} I_{S(X_i)\geq\gamma}} \tag{6}$$

Based on the above analysis, the main steps of the CE-based optimization algorithm for the CEPRMcan be summarized as a procedure composed of the following four steps:

1. Choose an initial parameter vector $\hat{p}_0$, and set the iterating counter $t = 1$;
2. Draw a random sample $X = \{X_1, ..., X_N\}$ from the probability density $f(\cdot; \hat{p}_{t-1})$. Calculate the value of $S(X_i)$ for all i, and then, sort them in descending order, having $S_{(1)} \leq ... \leq S_{(N)}$. Calculate the $(1-\rho)$-quantile of the current sample as:

$$\hat{\gamma}_t = S_{(\lceil (1-\rho)N \rceil)} \tag{7}$$

3. Solve for $\hat{p}_t = (\hat{p}_{(t,1)}, ..., \hat{p}_{(t,n)})$ via Equations (4) and (5), which means calculating the j-th success probability $\hat{p}_{(t,j)}$ through Equation (6), for all $j = 1, ..., n$;
4. If the stopping criterion is met, such as convergence or the maximum number of iterations is reached, then stop; otherwise, increase the counter t by one and start a new iteration from Step 2.

Remark 1 (Smoothed Updating)**.** In the above iterative procedure, *smoothed updating* is used instead of updating $\hat{p}_t$ directly from $\hat{p}_{t-1}$ through Equation (5). Thus, a smoothing coefficient $0 < \alpha < 1$ is used for updating the Bernouilli parameter as:

$$\hat{p}_t = \alpha \hat{p}_t + (1-\alpha)\hat{p}_{t-1} \tag{8}$$

This smoothed updating procedure performs better than a directly updating one, because it can prevent the situation that once zeros or ones occur in the parameter vectors, they often will remain so forever. Through extensive experiments, Boer *et al.* [19] found that, for optimization problems involving discrete random variables, a smoothing coefficient with a value between 0.4 and 0.9 would give the best results.

Remark 2 (Equivalent Objective Function)**.** The objective function in the admission control optimization problem **P1** is to maximize the total revenue within the available capacity constraints, while in the CE-based optimization problem, the objective function is not associated with constraints. Thus, transformations are needed for combining the objective function in Equation (1) and constraints in Equations (2) and (3) to get the new objective function $S(X)$, and we develop an equivalent objective function in CE as:

$$S(X) = \sum_{j=1}^{n} x_j p_{ite_j} + \sum_{j=1}^{n} p_{ite_j} * \sum_{i=1}^{m} min(c_i - \sum_{j=1}^{n} r_{i,ite_j} x_j, 0) \tag{9}$$

The second part in the right side of Equation (9) guarantees that, for a random generated solution, which violates the resource capacity constraints, the value of its revenue is negative, and thus, it will always have chances to be improved, *i.e.*, it will not be selected as the final best solution. On the other hand, for a feasible solution, its revenue value obtained from Equation (9) equals the one calculated from Equation (1).

Based on the above analysis, Algorithm 1 illustrates the whole CE-based optimization procedure for the VM admission control problem in pseudo-code.

Algorithm 1 The Cross-entropy-based VM admission control optimization algorithm

Input: L: List of the arrived VM service requests in a period
T: Set of VM instance types supported by the cloud provider
Output: S: The eligible set of VM service requests accepted and its revenue
1. set N as sample size, *maxits* as maximum iterations, *1-*ρ as quantile and d to indicate how many CE iterations to wait until declaring a solution is found; // Initialize parameters
2. int iterCounter=0, dCounter=0;
3. set $\widehat{p}_0$=(1/2,1/2,...,1/2) and dCounter=1; // set the initial parameter vector
4. set S_{best}=0, S_{old}=0; // set the value of best current solution and best solution in last iteration
5. WHILE ($iterCounter \leq maxits$ && $dCounter \leq d$)
// While maxits not exceeded and not converged
6. Draw a random sample $X_1, ..., X_N$ for Bernoulli vectors with probability vector $\hat{p}_{t-1}$;
7. Calculate the value of $S(X_i)$ for all i according to (Equation (9));
8. Sort all $S(X_i)$ in descending order, having $S_{(1)} \leq ... \leq S_{(N)}$;
9. if $(S_{old} == S_1)$
10. dCounter++;
11. else
12. dCounter=0;
13. end if;
14. Calculate the $1-\rho$-quantile of the current sample as $\hat{\gamma}_t$ according to (Equation (7));
15. Update pdf parameter vector $\widehat{p}_t$ with the best samples through (Equation (6));
// Update parameters of sampling distribution
16. if $(S_1 > S_{best})$
17. $S_{best} = S_1$;
18. end if;
19. $S_{old} = S_1$;
20. t=t+1; // Increment iteration
21. END WHILE
22. Return the Bernouilli parameter associated with the optimal solution to indicate the accepted set of VM requests and its performance value as the obtained revenue.

6. Experimental Evaluation

In order to validate the efficiency and effectiveness of the proposed admission control optimization algorithm, we implemented it as part of our public cloud computing platform. Experimental setups, baseline heuristics for comparison and numerical results of this implementation are explained next.

6.1. Experimental Settings

We consider a public cloud computing platform as shown in Figure 1, which consists of clusters of physical servers, and we assume that there are 30 VM instance types supported by the cloud provider. The configuration settings and prices of these 30 VM types are collected and generated based on the real-world popular public IaaS platforms, Amazon EC2 services [5] and Aliyun [18]. The number of virtual CPU cores required by each VM instance type is varied from one to 16, and the memory capacity required by each VM instance type is varied from 1 GB to 30 GB. Table 1 gives a sample of VM types that we used in our work. In order to evaluate the efficiency and effectiveness of our proposed admission control mechanism, we have varied the number of total resource capacities and the total arriving VM service requests to generate a large number of test cases. Since the effectiveness of an optimal admission control algorithm is more important for the cloud data centers with higher load factors (the ratio between the total capacity required and the total available capacity), for each test case,

the number of total available capacities and the total number of VM service requests are randomly set in a way that the load factors of the considered cloud system vary between 1.2 and 2. The number of VM service requests is varied from 40 to 200 with a step of 10. Additionally, the VM instance types requested by cloud users are selected randomly from the 30 VM types, following a normal distribution.

Table 1. Sample VM instance types.

No.	Number of CPU Cores	Memory	Cost ($ Per Hour)
1	1	1 GB	0.026
2	2	4 GB	0.18
3	4	16 GB	0.355
4	2	8 GB	0.20
5	4	8 GB	0.213
6	8	16 GB	0.397

The parameters of the proposed CE-based optimization algorithm for admission control are as follows:

- the sample size, *i.e.*, the number of Bernouilli vectors to generate in each iteration, is set as N=[200, 1000];
- the value of ρ for the quantile is 0.05;
- the value of the smoothing coefficient for updating Bernouilli parameters is set as $\alpha = 0.75$;
- the maximum number of iterations is 200;
- the degree to indicate how many CE iterations to wait before declaring a optimal solution is found is set as $d = 10$.

Additionally, in order to avoid biasing results due to randomness and improve the accuracy of the experimental results, for every experimental setting, the executions of our CE-based optimization algorithm are repeated 10 times to get average values, which are used as final results.

6.2. Reference Admission Control Heuristics

Since greedy approaches are generally used for solving optimization problems such as VM provisioning and allocation in clouds [11,21,22], in this paper, in order to test the effectiveness of our proposed CE-based VM admission control optimization approach and also propose a systematic approach to evaluate VM admission control algorithms in cloud data centers, based on the existing literature, we implement three greedy approaches with classic heuristics for VM admission control problems and compare the performance of our approach to theirs. The admission control heuristics implemented in this work used for comparison are described below:

- Highest revenue first (HRF): This heuristic always accepts the VM service requests with the largest revenue first. It first sorts all of the VM service requests by their revenue values associated with their requested VM instance types and then picks VM requests one by one from the sorted set, from highest to lowest, until no more services can be admitted due to there being not enough available resource capacities.
- Most profitable first (VRF): This heuristic always accepts the most profitable VM service requests first. The profitability of a VM service is calculated as the ratio between the revenue gained from it and the number of capacities needed. Then, all of the VM service requests are sorted and picked one by one based on these values of profitability. Since the VM instances considered in this work are heterogeneous, the profitability calculated along different resource dimensions would be different. In this paper, we use the number of CPU cores and the capacity of memory separately to calculate the profitability of VM requests, and these two heuristics are denoted as VRF_C (most profitable first by CPU) and VRF_M (most profitable first by memory), respectively.

6.3. Numerical Results and Discussion

In the following, we provide the numerical results of our experiments and illustrate the corresponding detailed settings. From the aspects of both efficiency and effectiveness, we discuss how our approach can outperform the reference three greedy approaches with classic heuristics and provide real-time performance guarantees in cloud computing environments.

6.3.1. Efficiency Validation

In this work, in order to verify the stability and scalability, as well as the efficiency of the proposed CE-based admission control optimization algorithm, we change the problem sizes in experiments to evaluate the average execution time and convergence of our proposed algorithm. We vary the problem size, *i.e.*, the number of VM service requests, from 40 to 200 with a step of 10 and examine the execution time and how many iterations the proposed algorithm needs before converging to an optimal solution. Besides, we also examine the effect of the sample size in the CE algorithm, by varying the sample size N from 200 to 1000 with a step of 50.

The distributions of iterations and run-time for these experiments are shown in Figures 2 and 3, respectively. It can be seen from Figure 2 that with the increasing of the number of VM requests and the decreasing of the sample size, the number of iterations needed by our algorithm before converging to an optimal solution is increasing slightly. As an example, Table 2 reports the execution trace of the CEVMAC procedure for a representative test case, by showing the best revenue value obtained so far in every iteration and its associated Bernoulli vector, which consists of the decision variables indicating whether to accept a VM request or not. Note that for all experimental settings, the CEVMAC algorithm converges quickly, and the number of iterations falls in the range of [10, 70]. Furthermore, considering the run-time required by our algorithm under different experimental settings, from Figure 3, we can see with the increasing of the number of VM requests that the run-time of our proposed optimization algorithm under the same sample size in CE algorithm increases slowly, but never exceeds 1 s. Accordingly, when keeping the number of VM requests unchanged, the run-time of our proposed optimization algorithm also increases to a small extent with the increasing of the sample size in the CE algorithm. Note that the run-times of all of these experiments with different problem sizes all fall below 1 s. Overall, we can conclude that the proposed CEVMAC algorithm is very efficient in finding an optimal solution for the VM admission control problem, since for the problems of reasonable sizes, the run-time required for reaching an optimal solution is at most 1 s, and the number of iterations required for convergence falls below 70. Thus, our proposed approach is suitable for on-line implementations and can provide real-time performance guarantees in cloud computing environments.

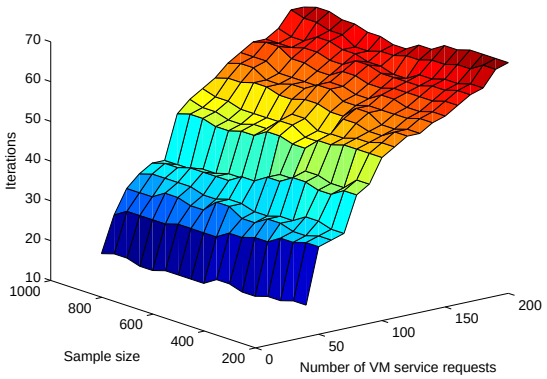

Figure 2. Iterations required by cross-entropy-based VM admission control optimization algorithm (CEVMAC) for reaching convergence under different experimental settings.

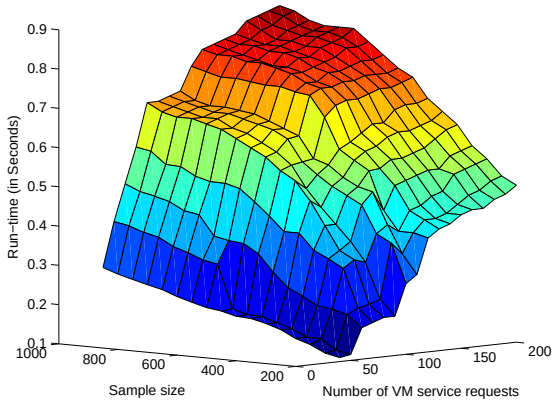

Figure 3. Run-time of CEVMAC on a 3.20-GHz i5-3470 CPU from Intel for different experimental settings.

Table 2. The execution trace in CEVMAC for a test case of VM admission control.

t	S_{best}	$\widehat{p}_t$
0		(0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5)
1	14.07	(1.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0)
2	14.11	(1.0 0.0 1.0 1.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.0 0.0 1.0 0.0)
3	14.17	(1.0 1.0 1.0 0.0 1.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 0.0 1.0 1.0 1.0 0.0 1.0)
4	14.17	(1.0 1.0 1.0 0.0 1.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 0.0 1.0 1.0 1.0 0.0 1.0)
5	14.30	(1.0 1.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0)
6	14.30	(1.0 1.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0)
7	14.30	(1.0 1.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0)
8	14.30	(1.0 1.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0)
9	14.30	(1.0 1.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0)
10	14.30	(1.0 1.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0)
⋮	⋮	⋮ ⋮ ⋮ ⋮ ⋮ ⋮ ⋮
14	14.30	(1.0 1.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0)

6.3.2. Effectiveness Evaluation

In order to evaluate the effectiveness of our proposed CE-based optimization algorithm for VM admission control, we have conducted several and extensive experiments to compare our algorithm to the reference admission control heuristics, which have been discussed in previous sections. In order to investigate the impact of the problem size on the effectiveness of the admission control algorithm, we vary the number of VM service requests from 50 to 200 with an increment of 50. Additionally, for each population of VM requests, the overall physical resource capacities are set accordingly, by keeping the value of the load factor as 1.2, 1.3, 1.4 and 1.5, respectively. Thus, for every population of VM requests, four test cases are generated, each with its own load factor and overall physical resource capacity. For the sake of clarity, we apply normalization to the experiment results of all test cases. Figure 4 demonstrates the normalized revenue obtained by the data centers using our CEVMAC and the other three heuristics, *i.e.*, HRF, VRF_C and VRF_M. It can be observed that the best results in all investigated problem instances are achieved by our CEVMAC approach. It can be seen that our CEVMAC algorithm can achieve improvement in revenue by up to 25% more than the other three heuristics. Thus, we say that our CEVMAC approach outperforms the other three heuristics for the VM admission control problem.

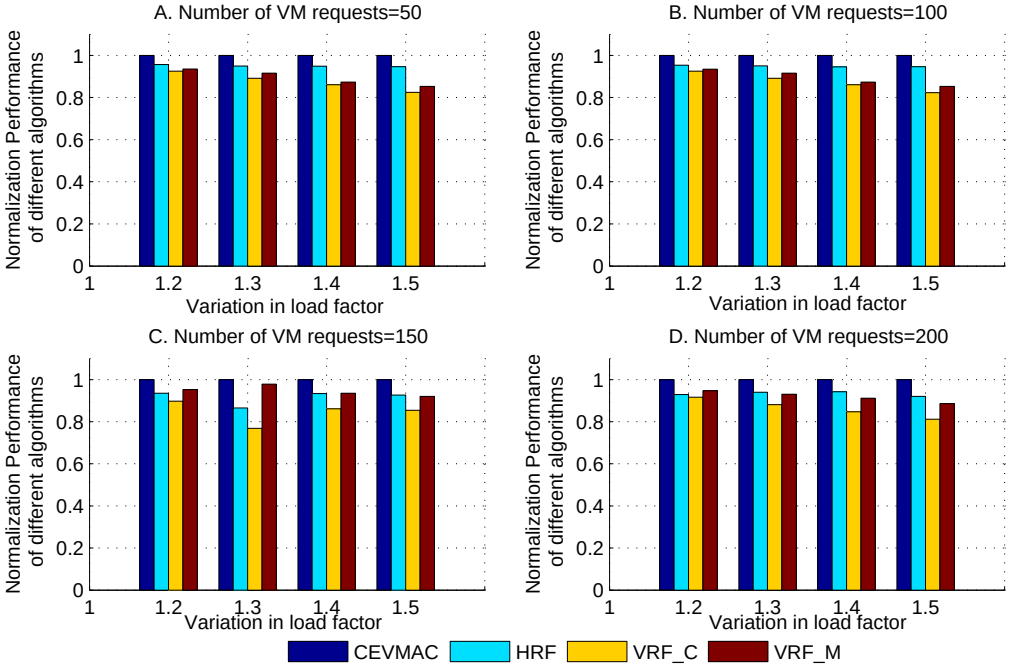

Figure 4. Normalized revenue of the data center with different algorithms, for different numbers of VM requests and load factors.

7. Conclusions

In this paper, we considered the revenue maximization problem during the VM service request admission control and placement process in public cloud data centers. Specifically, for addressing the heterogeneous VM admission control problem, which is formulated as an NP-hard multiple-dimensional knapsack problem, we proposed a cross-entropy-based optimization algorithm to maximize a cloud provider's revenue. The proposed approach can obtain near-optimal solutions to suggest an eligible set for the cloud provider to accept into its data centers, from the waiting VM service requests arriving in the system. Experimental results of a simulated environment demonstrated that our proposed cross-entropy-based admission control optimization algorithm is efficient and effective in maximizing cloud providers' revenue in public cloud computing environments. For the future, we are planning to investigate how to make VM admission control decisions and schedule VMs dynamically based on the prediction of the arrivals of IaaS requests to get optimal admission control and scheduling plans.

Acknowledgments: This work is supported in part by the National Natural Science Foundation of China (61402263), the Natural Science Foundation of Shandong Province (ZR2014FQ031) and the Science & Technology Development Projects of Shandong Province (2014GGH201007), China. The authors would like to thank the anonymous referees for their insightful comments that have significantly improved the quality of the paper.

Author Contributions: The work presented in this paper corresponds to a collaborative research by all authors, whereas Li Pan wrote the main paper. Pan Li and Datao Wang discussed the results and implications and commented on the manuscript at all stages. Both authors have read and approved the final manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Buyya, R.; Yeo, C.S.; Venugopal, S.; Broberg, J.; Brandic, I. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Gener. Comput. Syst.* **2009**, *6*, 599–616.
2. Armbrust, M.; Fox, A.; Griffith, R.; Joseph, A.D.; Randy, K. A view of cloud computing. *Commun. ACM* **2010**, *4*, 50–58.

3. VMware vCloud Suite. Available online: http://www.vmware.com/products/vcloud-suite/ (accessed on 11 March 2016).
4. OpenStack Open Source Cloud Computing Software. Available online: http://www.openstack.org/ (accessed on 11 March 2016).
5. Amazon Web Services. Available online: https://aws.amazon.com/ (accessed on 11 March 2016).
6. Li, K.; Brock, S.; Zheng, H.; Wu, J. Migration-based Virtual Machine Placement in Cloud System. In Proceedings of the IEEE 2nd International Conference on Cloud Networking (CloudNet), San Francisco, CA, USA, 11–13 November 2013; pp. 83–90.
7. Xiao, P.; Liu, B. An energy-efficient virtual machine scheduler with I/O collective mechanism in resource virtualisation environments. *Int. J. Netw. Virtual Organ.* **2013**, *4*, 311–326.
8. Zou, S.; Wen, X.; Chen, K.; Huang, S.; Chen, Y.; Liu, Y. VirtualKnotter: Online virtual machine shuffling for congestion resolving in virtualized data center. *Comput. Netw.* **2014**, *67*, 141–153.
9. Bazarbayev, S.; Hiltunen, M.; Joshi, K.; Sanders, W.H.; Schlichting, R. Content-Based Scheduling of Virtual Machines (VMs) in the Cloud. In Proceedings of the IEEE 33rd International Conference on Distributed Computing Systems, Philadelphia, PA, USA, 8–11 July 2013; pp. 93–101.
10. Luo, G.; Qian, Z.; Dong, M.; Ota, K.; Lu, S. Network-Aware Re-Scheduling: Towards Improving Network Performance of Virtual Machines in a Data Center. *Algorithms Arch. Parall. Process.* **2014**, *8630*, 255–269.
11. Wang, Q.; Ren, K.; Meng, X. When cloud meets eBay: Towards effective pricing for cloud computing. In Proceedings of the IEEE INFOCOM, Orlando, FL, USA, 25–30 March 2012; pp. 936–944.
12. CloudSim: A Framework For Modeling and Simulation of Cloud Computing Infrastructures and Services. Available online: http://www.cloudbus.org/cloudsim/ (accessed on 11 March 2016).
13. Jin, H.; Gao, W.; Wu, S.; Shi, X.; Wu, X.; Zhou, F. Optimizing the live migration of virtual machine by CPU scheduling. *J. Netw. Comput. Appl.* **2011**, *4*, 1088–1096.
14. Razali, R.A.M.; Rahman, R.A.; Zaini, N.; Samad, M. Virtual machine migration implementation in load balancing for Cloud computing. In Proceedings of the 5th International Conference on Intelligent and Advanced Systems (ICIAS), Kuala Lumpur, Malaysia, 3–5 June 2014; pp. 1–4.
15. Shrivastava, V.; Zerfos, P.; Lee, K.W.; Jamjoom, H. Application-aware virtual machine migration in data centers. In Proceedings of the 2011 IEEE INFOCOM, Shanghai, China, 10–15 April 2011; pp. 66–70.
16. Bose, S.K.; Brock, S.; Skeoch, R.; Rao, S. CloudSpider: Combining Replication with Scheduling for Optimizing Live Migration of Virtual Machines across Wide Area Networks. In Proceedings of the 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), Newport Beach, CA, USA, 23–26 May 2011; pp. 13–22.
17. Gaetano, F.A.; Pietro, C.; Patrizio, D.; Alberto, G.; Matteo, M.; Andrea, R. A Hybrid Cross-Entropy Cognitive-Based Algorithm for Resource Allocation in Cloud Environments. In Proceedings of the IEEE Eighth International Conference on Self-Adaptive and Self-Organizing Systems (SASO), London, UK, 8–12 September 2014; pp. 11–20.
18. Aliyun Cloud Services. Available online: https://www.aliyun.com/ (accessed on 11 March 2016).
19. Rubinstein, Y.R. Optimization of Computer simulation Models with Rare Events. *Eur. J. Oper. Res.* **1997**, *1*, 89–112.
20. Rubinstein, Y.R.; Kroese, D.P. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, and Machine Learning*; Springer-Verlag: New York, NY, USA, 2004.
21. Nejad, M.M.; Mashayekhy, L.; Grosu, D. Truthful Greedy Mechanisms for Dynamic Virtual Machine Provisioning and Allocation in Clouds. *IEEE Trans. Parall. Distrib. Syst.* **2014**, *2*, 594–603.
22. Ma, L.; Lu, Y.; Zhang, F.; Sun, S. Dynamic Task Scheduling in Cloud Computing Based on Greedy Strategy. *Trustworthy Comput. Serv.* **2013**, *1*, 156–162.