Mobile Netw Appl (2018) 23:227–238
https://doi.org/10.1007/s11036-017-0925-7

Improving Resource Utilization via Virtual Machine Placement in Data Center Networks

Tao Chen[1] · Yaoming Zhu[1] · Xiaofeng Gao[1] · Linghe Kong[1] · Guihai Chen[1] · Yongjian Wang[2]

Published online: 25 September 2017

Abstract The resource utilization of servers (such as CPU, memory) is an important performance metric in data center networks (DCNs). The cloud platform supported by DCNs aims to achieve high average resource utilization while guaranteeing the quality of cloud services. Previous papers designed various efficient virtual machine placement schemes to increase the average resource utilization and decrease the server overload ratio. Unfortunately, most of virtual machine placement schemes did not contain the service level agreements (SLAs) and statistical methods. In this paper, we propose a correlation-aware virtual machine placement scheme that effectively places virtual machines on physical machines. First, we employ neural networks model and factor model to forecast the resource utilization trend data according to the historical resource utilization data. Second, we design three correlation-aware virtual machine placement algorithms to enhance resource utilization while meeting the user-defined SLAs. The simulation results show that the efficiency of our virtual machine placement algorithms outperforms the generic algorithm and constant variance algorithm by about 15%-30%.

Keywords Virtual machine placement · Prediction · Correlation · Data center networks

✉ Xiaofeng Gao
gao-xf@cs.sjtu.edu.cn

Tao Chen
tchen@sjtu.edu.cn

Yaoming Zhu
grapes_islet@sjtu.edu.cn

Linghe Kong
linghe.kong@sjtu.edu.cn

Guihai Chen
gchen@cs.sjtu.edu.cn

Yongjian Wang
wangyongjian@stars.org.cn

[1] Shanghai Key Laboratory of Scalable Computing and Systems, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, 200240, China

[2] Key Laboratory of Information Network Security of Ministry of Public Security, The Third Research Institute of Ministry of Public Security, Shanghai, China

1 Introduction

Data center networks (DCNs), the essential backbone infrastructure of cloud services such as cloud computing, cloud storage, and cloud platforms, attract increasing attentions in both academia and industry. Cloud data centers aims to provide an integrated platform with a pay-as-you-go business model to benefit tenants at the same time, which is gradually adopted by the mainstream IT companies, such as Amazon EC2, Google Cloud Platform and Microsoft Azure. The cloud services are provided by the virtualization on shared resources and utilities in DCNs, such as CPU, memory, and bandwidth. Various tenants buy virtual machines (VMs) within a certain period of time to run their applications [3]. Owing to multi-tenant demands, all kinds of workloads physically coexist but are logically isolated in DCNs, including data-intensive and latency-sensitive services, search engines, business processing, social-media networking, and big-data analytics [18]. Elastic and dynamic resource provisioning is the basis of DCN performance, which is achieved by virtualization technique to decrease the cost of leased resources and to

increase resource utilization in cloud platforms. Therefore, the effectiveness of virtualization directly determines the performance of DCNs.

The design goal of a DCN is originally to meet the peak workloads of tenants. However, at most time, DCNs are suffering from high energy cost due to low server utilization. A large number of servers are running with low workloads while consuming almost the same amount of energy as servers with high workloads. The cloud service providers have to spend more money on cooling bills to keep the servers in normal running. Allocating resources in an energy-effective way while guaranteeing the Service Level Agreements (SLAs) for tenants is essential for them.

Pervious literatures focused on enhancing the average utilization without violating SLAs. Some researchers focus on fair allocation schemes. Bobroff et al. [4] proposed a virtual machine placement (VMP) scheme to dynamically manage SLA violations, which forecasts the future resource demands and calculates the prediction error. Nevertheless, the VMP scheme takes no correlation between VMs into consideration, and only predicts demand of a single VM. In [22], the authors proposed joint-VM-provisioning, which can achieve 45% improvements in terms of overall resource utilization compared to VM-by-VM provisioning.

In this paper, we present a correlation-aware VMP scheme that effectively places virtual machines on physical machines. First, we employ Neural Networks model and factor model to forecast the resource utilization trend according to the historical resource utilization data. Second, we design three correlation-aware placement algorithms to enhance resource utilization while meeting the user-defined service level agreements. The results of extensive simulations showed that the efficiency of our VMP scheme outperforms the previous work by about 15%-30%.

The rest of the paper is organized as follows. Section 2 introduces the related work about resource demand prediction, virtual machine placement, virtual machine migration, and data center networks. Section 3 proposes the correlation-aware virtual machine placement system. Section 4 concludes this paper.

2 Related work

2.1 Resource demand prediction

It is probable to mitigate hot spots in DCNs by appropriate prediction schemes. Demand prediction methods will provide us early warnings of hot spots. Hence, we can adopt measures to ease the congestions in DCNs and allocate resource in a way that guarantee the performance of applications for tenants. The demand prediction methods usually fall into time series and stochastic process analyzes.

The ARIMA model is often used to predict time series data. In [4], the researchers forecast the future demand and propose the prediction error model. Nevertheless, they take no correlation between VMs into consideration, and only predict demand of a single VM. Lin et al. accurately predicts the future VM workloads by seasonal ARIMA models [21]. Qiu et al. employs SARMA model on Google Cluster workload data to predict future demand consumption [25]. In [26], the authors use a variant of the exponentially weighted moving average (EWMA) load predictor. For workloads with repeating patterns, PRESS derives a signature for the pattern of historic resource utilization, and uses that signature in its prediction. PRESS uses a discrete-time Markov chain with a finite number of states to build a short-term prediction of future metric values for workloads without repeating pattern, such as CPU utilization or memory utilization [11]. In [16], Markov chain model is applied to capture the temporal correlation of VM resource demands approximately. In [35], an adaptive prediction algorithm was designed to improve TRE efficiency through dynamically adjusting the prediction window size based on the hit ratio of historical predictions.

2.2 Virtual machine placement

Virtual Machine Placement (VMP) problem involves mapping virtual machines (VMs) to physical machines (PMs). A proper VM mapping scheme can lead to less number of PMs required and lower energy cost. A poor resource allocation scheme may require more PMs and may induce more service level agreement (SLA) violations. In [4], the authors proposed a VMP scheme to manage SLA violations. A method was designed to identify servers which benefit most from dynamic migration. In [22] the authors proposed joint-VM-provisioning, which can achieve 45% improvements in terms of overall resource utilization compared to VM-by-VM provisioning. They first introduced an SLA model that maps application performance requirements to resource demand requirement. Kim et al. [19] proposed a novel correlation-aware virtual machine allocation for energy-efficient datacenters. Specifically, they take correlation information of core utilization among virtual machines into consideration. Wang et al. [27] attempt to explore particle swarm optimization (PSO) to minimizing the energy consumption. They design an optimal VMP scheme with the lowest energy consumption. In [20], authors propose a VMP scheme which minimizes the energy consumption of

Table 1 Summary of VMP schemes

Schemes	Correlation-aware	Energy-aware	Prediction	SLA
Bobroff et al. [4] (2007)				⋆
Meng et al. [22] (2010)	⋆	⋆		⋆
Kim et al. [19] (2013)	⋆	⋆		⋆
Wang et al. [27] (2013)		⋆		⋆
Khalilzad et al. [20] (2015)		⋆		⋆
Chen et al. [8] (2016)	⋆	⋆	⋆	⋆

the data center by consolidating VMs in a minimum number of PMs while respecting the latency requirement of VMs. In [8], the authors design correlation-aware algorithms to solve VMP problem.

We summarize the VMP schemes in Table 1, including the properties of correlation-aware, energy-aware, prediction, and SLA. If a VMP scheme has a property, the corresponding cell is filled with a ⋆.

2.3 Virtual machine migration

Virtual Machine Migration, a sort of VM replacement, aims to satisfy different resource management objectives (e.g., load balancing, fault tolerance, power management). Live migration is the technology that makes VM replacement realistic with a downtime of only 60ms.

Nelson et al. [23] proposed a fast and transparent application migration system. The author claimed that this is the first system that can migrate unmodified applications on unmodified mainstream operating systems. They did experiments and measured that for a variety of workloads, application downtime caused by migration is less than a second. Clark et al. [9] presented the design, implementation and evaluation of high-performance OS migration built on top of the Xen virtual machine monitor. OSes continue to run when migrating a large quantities of contents. Downtime can be reduced to as low as 60ms. This new technology is a powerful tool for server consolidation. Wood et al. [29] implemented a black-box approach that is fully OS- and application-agnostic and a gray-box approach that exploits OS-level and application-level statistics. They designed a hot spot detection algorithm that determines

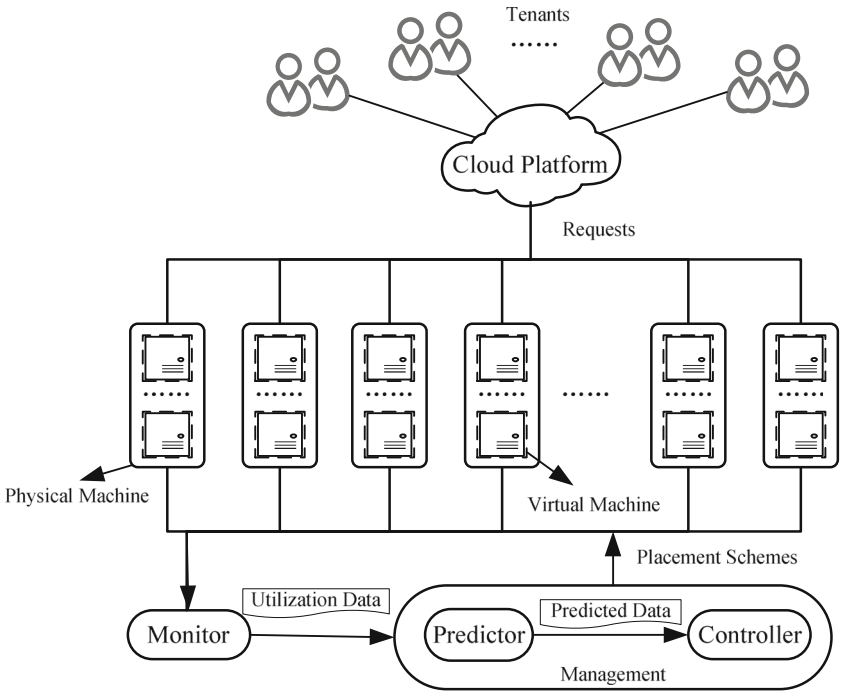

Fig. 1 VM Placement system architecture

when to migrate virtual machines. They also implemented a hot spot mitigation algorithm that decides what and where to migrate and how much to allocate after the migration.

Ye et al. [32] compared the live migration efficiency with different resource reservation approaches and proposed corresponding optimization methods. Ghorbani et al. [10] presented LIME architecture, which can efficiently migrate a collection of virtual machines and virtual switches, for any arbitrary Software-Defined Network (SDN) controller and applications. Xu et al. [31] proposed iAware, a lightweight interference-aware VM live migration strategy, which jointly estimates and minimizes both migration and co-location interference among VMs by a multi-resource demand-supply model. Wood et al. [30] present the CloudNet architecture, which provides optimized support for live WAN migration of VMs. A set of optimizations that minimize the cost of transferring storage and VM memory during migrations over low bandwidth links are also proposed. In [5], the authors proposed NICE, a network-aware VM consolidation scheme, to save the energy cost of DCNs. Yu et al. [34] presented a stochastic load balancing scheme for virtual machine migration to minimize the migration cost and guarantee the lower resource overload. In [33], an optimal allocation algorithm was designed to achieve virtual machine migration while minimizing the total migration cost for the virtual cluster scaling.

2.4 Data center networks

In a data center, the data center network (DCN) is an essential part, which interconnect the physical components (e.g., servers, switches) in a specific topology with cables and optical fibers. In the process of virtual machine placement or migration, DCNs also determine the period and efficiency.

According to [7], the topologies can be categorized into switch-centric, server-centric, and enhanced topologies. In switch-centric topologies, the switches are enhanced to satisfy networking and routing requirements, while the servers are almost all unmodified, such as Fat-Tree [1], VL2 [12]. In server-centric topologies, servers are modified to be responsible for networking and routing, while commodity switches without modification are employed only for forwarding function, such as DCell [13], BCube [14]. DCNs can be enhanced by optical devices and wireless antennas, such as OSA [6], 3D beamforming [36]. More details can be referred to the survey on DCNs [7].

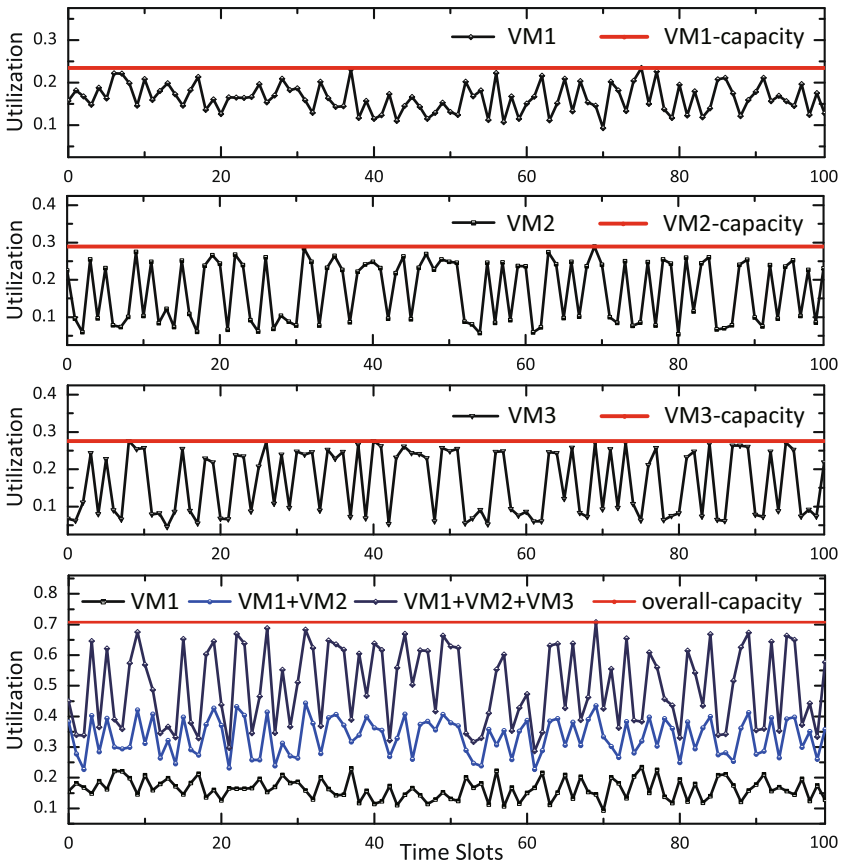

Fig. 2 An example of the correlation between three VMs

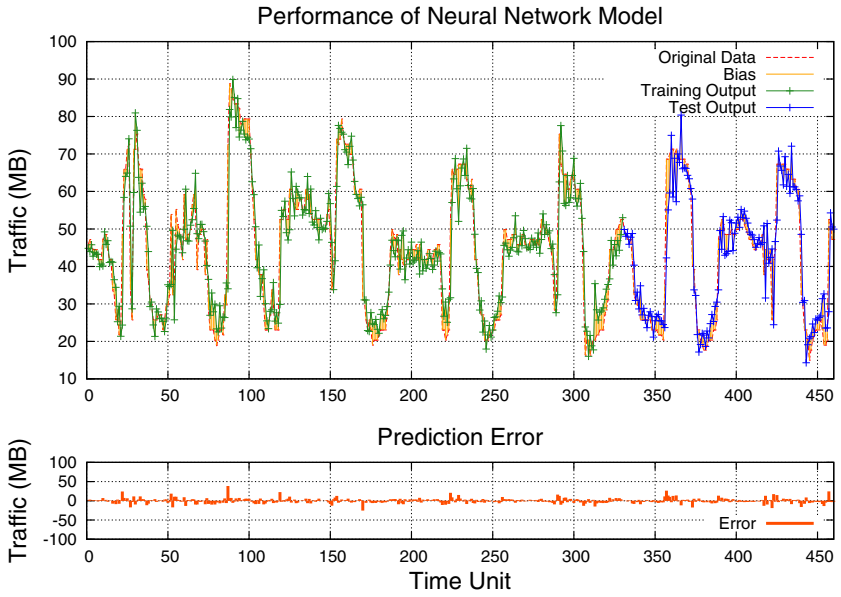

Fig. 3 Performance of NARNET

3 Correlation-aware virtual machine placement scheme

3.1 System architecture

We propose a correlation-aware virtual machine placement (VMP) system for data center networks (DCNs) that predicts the future resource demand/utilization of requests and minimize the number of physical machines (PMs). The VMP system considers the correlations between virtual machines (VMs), and satisfies a user-defined server level agreement (SLA) at the same time.

The main modules of our VMP system, i.e., monitor, predictor and controller, are shown in Fig. 1. Tenants submit resource requests to the cloud platform. The cloud platform allocates the resources (VMs) for the requests. VMs are usually hosted on PMs in DCNs. Monitor module records the historical utilization data of VMs and transmit it to Predictor module. The predicted data generated from Predictor is delivered to Controller modular that makes a strategic decision for VM placement problem. An new VM placement strategy happens periodically every 100 time slots (a resource demand data recorded at a time slot).

Traditionally, a VMP scheme placed a VM at a time. In [22], the authors argued that the anti-correlation between VMs can be utilized. They placed two VMs at a time and allocated as less resource as possible for the two VMs. However, the negative correlation between three VMs also can be leveraged, as shown in Fig. 2. By analogy, the joint-provisioning of any number of VMs without SLA violations can be done. The overall capacity allocated for the three VMs under joint-provisioning is about 70% of a PM, while the traditional VM placement needs to allocate about 85% capacity for the same three VMs.

3.2 Prediction

3.2.1 Neural networks

In [28], the authors applied ARIMA and GARCH model to forecast the trend and volatility of the future demand. ARIMA performs well when an initial differencing step can be applied to remove non-stationarity. However, ARIMA is a linear time series model and may not work otherwise.

Table 2 A simple data set of factor model

Time	0	1	2	3	4	5	6	7	8	9	10
VM0	10	20	30	20	10	25	20	15	10	5	7
VM1	40	45	39	34	28	15	23	14	7	15	14
VM2	30	33	31	35	32	29	31	33	36	41	33
VM3	50	45	47	54	61	49	36	28	32	40	38
VM4	10	13	16	18	15	13	8	15	20	17	11
VM5	38	31	41	52	42	39	41	43	46	31	34
VM6	20	26	31	25	31	23	28	32	34	30	23
VM7	15	23	11	15	22	25	21	13	26	21	24
VM8	10	21	31	25	13	22	31	23	33	29	33
VM9	33	32	31	34	31	29	25	23	27	31	32

Neural Networks can be applied to predicted both linear and non-linear time series. For example, nonlinear autoregressive neural network (NARNET) can be trained to predict a time series from historical demand data.

Let NARNET(ni, nh) denotes a nonlinear autoregressive neural network with ni inputs and nh outputs. Such a model can be described as

$$U_i(t) = F(U_i(t-1), U_i(t-2), \ldots) + \varepsilon \quad (1)$$

where U_t is the variable of interest, and ε is the error term. We can the use this model to predict the value of U_{t+k}.

The performance of NARNET(10,20) is shown in Fig. 3. The simulation results shows that NARNET can predict future resource demand accurately.

3.2.2 Factor model

Wei et al. [28] used ARIMA model to predict resource demand series for each VM. However, each VM needs to train by ARIMA model, and it is not feasible for a large scale of VMs. For a mega production data center, ARIMA may not be able to work appropriately. In addition, ARIMA has poor performance when the resource utilizations of VMs have no obvious patterns, which may be obscured by various random factors.

We employ a factor model to solve these problems [24]. The demand series $\{U_i(t)\}$ of each VM i can be decomposed to several uncorrelated underlying factors $C_1(t), \ldots, C_Q(t)$ with a zero mean error term $\varepsilon(t)$:

$$U_i(t) = \alpha_{i1}C_1(t) + \ldots + \alpha_{iQ}C_Q(t) + \varepsilon(t), \forall i. \quad (2)$$

We first calculate $\alpha_{i1}, \ldots, \alpha_{iQ}$ effectively, then predict future demand at time t by predicting factor movements. Principal component analysis (PCA) is used to find underlying factors. Given P utilization series $\{U_1(t)\}, \ldots, \{U_P(t)\}$, PCA leverages an orthogonal transformation to the P utilization series to get a relatively small number of uncorrelated time series $\{C_1(t)\}, \ldots, \{C_Q(t)\}$, which are the principal components.

PCA is executed for all the VMs. Nevertheless, the dependencies on each factor of VMs are different. To predict the mean $\mu(t)$ and covariance $\Sigma(t)$ for utilization series $\{U_i(t) : i = 1, \ldots, P\}$ at time t, we first forecast the principal components $C_1(t), \ldots, C_Q(t)$ based on historical data, and then predict the error series $\varepsilon(t)$ to obtain its mean and error variance. Thus, we can predict $\mu_i(t)$ as

$$\mu_i(t) = \alpha_{i1}(t)C_1(t) + \ldots \alpha_{iQ}(t)C_Q(t) + \varepsilon(t), \forall i. \quad (3)$$

We illustrate the idea of factor analysis by a simple example. There are a data set as shown in Table 2.

We first do PCA to this simple data set to get eleven principal components. As the data set proceed, it is shown that the first three principal component accounts for 92.38% of the variance. Thus, we can select three principal components and just do ARIMA and GARCH for them. After choosing these three factors, we reconstruct the original data with errors to some extent. Suppose mu is the mean value of each attribute of the raw data, pc is the matrix of principal components and $score$ is the principal components scores, that is, the representation of the original data in the principal components space. We reconstruct the original data using principal components pc and principal components score $score$ by adding mu to the combination $pc \times score^T$. We use ARIMA and GARCH to forecast their mean and variance. After getting the mean and variance, we can reconstruct each VM's profile. With the profile and the original data, we can get the error term. As for the error term, we can use ARIMA to model the mean and use GARCH to model its variance. The error term in this example is rather small at the scale of 10^{-13}.

We can use ARIMA and GARCH to predict the future mean and variance of the first three principal components. Using the future mean and variance, together with the principal component scores, we can predict VMs' future demand (do not forget to add the error term to the data).

In [28], we have to do ten ARIMA and GARCH analysis to get the future mean and variance. By factor analysis, we can reduce the number ten to four. We only have to forecast the mean and variance of the largest three principal components and the mean and variance of the error term. The factor analysis really reduces the number of computations.

Next, let's look at a larger example. We generate 100 VM resource utilization traces and each trace contains 100 data. The first half are used to train the ARIMA model, and the other half are used for the testing. Each trace is generated using normal distribution with random mean value and variance.

By the traditional method called individual prediction used in [28], we had to train a separate ARIMA model and GARCH model for each virtual machine.

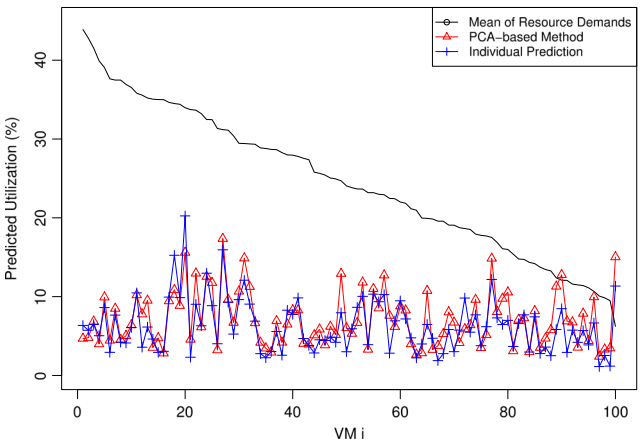

Fig. 4 Root mean square error of PCA-based method compared with individual prediction

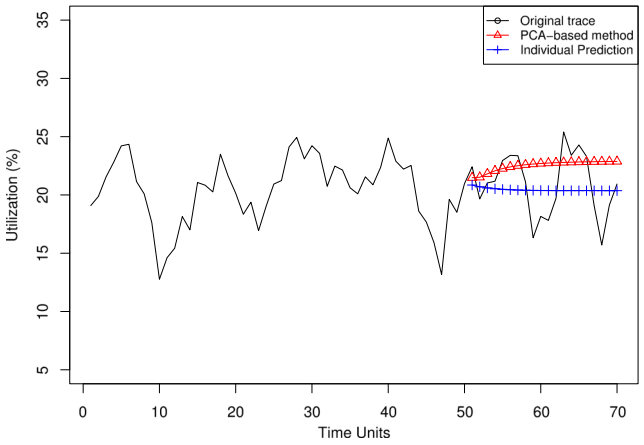

Fig. 5 20-step-ahead prediction performance of PCA-based method and individual prediction

As for PCA-based method, we pick 10 principal components, and train ARIMA and GARCH model to predict mean and variance for the next 10 time units for these 10 principal components respectively. The simulation results show that 10 principal components accounts for 60% of all the variance. Figure 4 shows that individual prediction and PCA-based method has similar performance which really set our mind at rest. We sort the VM id according to the average size of each VM. For VMs who have larger size, PCA-based algorithm has good performance while for small VMs both individual prediction and PCA-based algorithm will experience severe performance degradation.

Figure 5 shows us that the performance of PCA-based method compared with individual prediction method. We can see that time series analysis is no panacea. The 20-step-ahead prediction can not provide very accurate prediction, which only serve as an indicator to the trend.

In summary, PCA-based method can save us a lot of time instead of doing heavy computation of ARIMA model. The PCA-based method may work better and 10 principal components obtained may accounts for 80% of the total variance.

Table 3 Main symbols and descriptions

Symbol	Description
$V = \{v_1, \cdots, v_n\}$	Set of VMs
$S = \{s_1, \cdots, s_m\}$	Set of PMs
N_{PM}	Number of used PMs for placement
D_m	sum of resource demands in PM m
C	capacity of a PM
$\overline{o}$	overload ratio
$\overline{D}$	average resource demand (utilization)
ϵ	user-defined SLA
$E[D_m]$	sum of expectations of resource demands of all VMs on PM m
$var[D_m]$	variance of the workload with correlations between VMs on PM m

3.3 Virtual machine placement algorithms

In this subsection, we propose correlation-aware virtual machine placement (VMP) algorithms. The allocated resource for VMs should match the future resource demand to achieve high resource utilization of PMs while meeting user-defined SLAs. Table 3 summarizes the main symbols used in this paper.

We use two performance metrics, overload ratio $\overline{o}$ and average resource demand $\overline{D}$, to evaluate the effectiveness of our proposed VM placement algorithms. The former is the ratio of the number of time slots when the actual resource demand of a PM is higher than its capacity over all the time slots $\times$ N_{PM}. The latter is the average resource utilization of PMs over all the time slots. The objective of algorithms is to achieve low overload ratio $\overline{o}$ and high average resource utilization $\overline{D}$. We monitor resource demand (e.g., CPU or memory) for each VM and forecast conditional mean μ and the conditional variance σ. According to resource demand time series data, the correlation ρ between different VMs placed on the same PM is also calculated.

The correlation-aware VMP problem is a problem that minimizing the number of PMs and satisfying the SLAs in a correlation-aware VM placement way, which can be formulated as follows:

$$\min N_{PM} \tag{4}$$

$$s.t. \quad \Pr(D_m > C) < \epsilon, \ \forall m, \tag{5}$$

$$\sum_m x_{mn} = 1, \ \forall n, \tag{6}$$

$$x_{mn} \in \{0, 1\}, \ \forall m, \ \forall n. \tag{7}$$

The binary variable x_{mn} indicates VM n is hosted on PM m or not. D_m denotes the resource demand of VMs on PM m. C means the capacity of a PM. $\epsilon > 0$ is a small constant, called *user-defined SLA*, which means the maximum value of the overload ratio, that is, the cloud platform should guarantee the overload ratio under ϵ for tenants.

Equation 5 can be transformed to:

$$C \geqslant E[D_m] + c_\epsilon(0, 1)\sqrt{var[D_m]}$$

$$E[D_m] = \mu_1 x_{m1} + \mu_2 x_{m2} + \ldots + \mu_n x_{mn},$$
$$var[D_m] = \sum_{i,j} \rho_{ij}\sigma_i\sigma_j x_{mi} x_{mj}.$$

where $c_\epsilon(0, 1)$ is the $(1 - \epsilon)$-percentile of standard normal distribution with mean 0 and variance 1. For example, when $\epsilon = 2\%$, $c_\epsilon(0, 1) = 2.06$. $E[D_m]$ is the sum of expectations of resource demands of all VMs placed on PM m, and $var[D_m]$ is the variance of the workload with correlations between VMs taken into consideration.

After problem formulation, we propose our VMP algorithms. The first algorithm is *Correlation-Aware First-Fit* algorithm. The algorithm is similar to first-fit algorithm in solving the bin-packing problem, which is shown in Algorithm 1.

Algorithm 1 Correlation-aware First-Fit VMP Algorithm

Input: Historical resource demand data of VMs from the monitor.
Output: A VMP scheme with a user-defined SLA.

```
foreach VM n do
    foreach PM m do
        Add VM n to PM m;
        x_mn = 1;
        if E[D_m] + c_ε(0, 1)√var[D_m] < C then
            break;
        else
            Remove VM n from PM m;
            x_mn = 0;
        end
    end
end
```

Algorithm 1 is a first-fit algorithm which will place a certain VM into the first PM that can hold it with a certain probability less than a user-defined SLA. Since this problem is very similar to first-fit algorithm of bin packing problem, we can easily reach the inequality the number of PMs used by first-fit described above is no more than $2\times$ optimal number of PMs. If we first sort the VMs by the size, then this is very similar to first fit decreasing algorithm in bin packing problem. It has been shown to use no more than $\frac{11}{9}\mathbf{OPT} + 1$ bins (where **OPT** is the number of bins given by the optimal solution).

The second algorithm is *Correlation-Aware Best-Fit* algorithm, as shown in Algorithm 2. The "best-fit" idea is: at each time, placing a VM to the PM that has minimal free capacity and can accommodate it.

Algorithm 2 Correlation-Aware Best-Fit VM Placement Algorithm

Input: Historical resource demand data of VMs from the monitor.
Output: A VM placement scheme with a user-defined SLA.

```
foreach VM n do
    i = 1;
    Try to place VM n to the PM that has i-th
    minimum free capacity;
    x_mn = 1;
    if E[D_m] + c_ε(0, 1)√var[D_m] < C then
        break;
    else
        Remove VM n from PM m;
        x_mn = 0;
        i = i + 1;
    end
end
```

The third algorithm is *Correlation-Aware Minimum Bin Slack* (MBS) algorithm, as shown in Algorithm 3.

The MBS algorithm is a heuristic algorithm, in which each packing is determined in a search procedure that tests all possible subsets of VMs on the list which fit the PM capacity [15]. For each PM, we choose the subset of VMs with the minimum PM slack (remaining resource capacity) when placing them in the PM. If the algorithm finds a subset of VMs that fills the PM completely, the search is stopped, for there is no better packing possible.

Algorithm 3 Correlation-Aware Minimum Bin Slack VM Placement Algorithm

Input: Historical resource demand data of VMs from the monitor.
Output: A VM placement scheme with a user-defined SLA.

```
Sorting resource demand data of VMs in
non-increasing order
foreach PM m do
    i=1;
    Try to place i-th largest VM on PM m for
    achieving minimum slack;
    x_mn = 1;
    if E[D_m] + c_ε(0, 1)√var[D_m] < C then
        break;
    else
        Remove VM n from PM m;
        x_mn = 0;
        i=i+1;
    end
end
```

We compare our VM placement algorithms with the following benchmark algorithms:

Generic Algorithm (GA) A genetic algorithm is a search heuristic algorithm that is often used to solve NP problem. The VMP problem can be reduced to bin packing problem, and usually we can not find an optimal solution by using first fit algorithm and minimum bin slack algorithm.

Genetic Algorithm is an algorithm that mimics the process of evolution. It is often applied to generate solutions for optimization and search problems. Usually, a Genetic Algorithm will use techniques such as inheritance, mutation, selection, and crossover. We introduce the outline of the Genetic Algorithm algorithm as follows [17]:

1. Generate initial solutions and set g equals 1. The variable g represents the generation number.
2. Select two genotypes p_1 and p_2 randomly from the old generation.
3. Generate two offspring o_1, o_2 using crossover operation.

4. Do mutation operation to p_1, p_2 and generate offspring o_3 and o_4.
5. Select two best solutions from p_1, p_2, o_1, o_2, o_3, and o_4. Remove p_1 and p_2 from the population and add the solutions selected to the population.
6. if g equals G, terminate the Genetic Algorithm. If not set g equals $g + 1$ and return to step 2.

- Individual. In the Genetic Algorithm, a set of VM numbers in one PM correspond to a gene. Solutions generated by crossover operation and mutation operation do not depend on the sequence of genes.
- Initialization. Although Genetic Algorithm does not care about the quality of initial solutions, and will converge for almost all kinds of initial solutions. However, good initialization will reduce the number of iterations required to reach a optimal solution. Here, we use first fit to generate the first generation. We randomize the sequence of VMs to generate different initial solutions. Diversity is very important for the process of initial solution generation. One thing to note is that it is unlike first fit decreasing where VMs are sorted in the order of non-increasing size.

  We use fisher-yates shuffle to randomize the sequence of VMs. The algorithm is described in Algorithm 4. It is extremely fast, and is unbiased, so that every permutation is equally likely.
- Crossover Operation. The crossover operation is crucial to the whole Genetic Algorithm. It should guarantee that the offspring of two parents preserve or inherit good traits in VM placement problem. We use minimum bin slack algorithm to help us find better solution in the phase of crossover.

Algorithm 4 Fisher-Yates shuffle

Input: R is the array you want to shuffle
r is the size of the array
Output: A new permutation

```
for i = r − 1; i >= 1; − − i do
    j is a random integer ∈ [0, i];
    swap R's i^th element and R's j^th element;
end
```

Two parents p_1 and p_2 will produce their offspring o_1 in a way as follows:

1. Some PMs in p_1 are selected at random, and the VMs in them are copied directly to o_1.
2. In p_2, VMs allocation in the PMs where no VM are assigned in o_1 are copied to o_1.
3. VMs that are not assigned will be allocated space using first fit decreasing or minimum bin slack algorithm. VMs that belongs to the same PM in the past will be considered as one VM. We will see what it means in the following example.

Table 4 Number of used PMs, the overload ratio ($\overline{o}$), and average resource utilization of PMs ($\overline{D}$) in different time slots under different user-defined SLAs

	Time Slots 100-200			Time Slots 200-300			Time Slots 300-400		
$\epsilon = 2\%$	N_{PM}	$\overline{o}$	$\overline{D}$	N_{PM}	$\overline{o}$	$\overline{D}$	N_{PM}	$\overline{o}$	$\overline{D}$
GA	121	11.2%	57.65%	120	11.7%	58.04%	119	10.43%	57.76%
CV	109	0.13%	58.77%	108	0.1%	58.24%	110	0.06%	58.24%
FF	91	1.1%	70.45%	91	1.1%	70.44%	91	0.77%	70.44%
BF	91	1.1%	70.45%	91	1.1%	70.44%	91	0.79%	70.44%
MBS	90	1.3%	71.18%	91	0.7%	70.40%	89	0.95%	71.98%
$\epsilon = 5\%$									
GA	118	9.4%	58.63%	120	11.53%	58.14%	119	8.22%	57.75%
CV	97	2.8%	66.04%	97	3.1%	66.04%	97	2.4%	66.05%
FF	86	3.0%	74.49%	86	2.9%	74.49%	87	2.8%	73.63%
BF	86	3.2%	74.49%	86	3.4%	74.49%	87	2.4%	73.63%
MBS	84	4.3%	76.26%	84	3.5%	76.26%	84	3.6%	76.26%
$\epsilon = 10\%$									
GA	116	10.5%	59.64%	122	8.5%	57.90%	122	10.39%	57.65%
CV	90	4.5%	71.18%	90	4.7%	71.18%	90	4.1%	71.18%
FF	81	7.6%	79.09%	82	6.9%	78.12%	82	6.6%	78.12%
BF	81	7.7%	79.09%	82	6.8%	78.12%	82	6.7%	78.12%
MBS	80	9.2%	80.08%	80	8.4%	80.08%	80	8.9%	80.08%

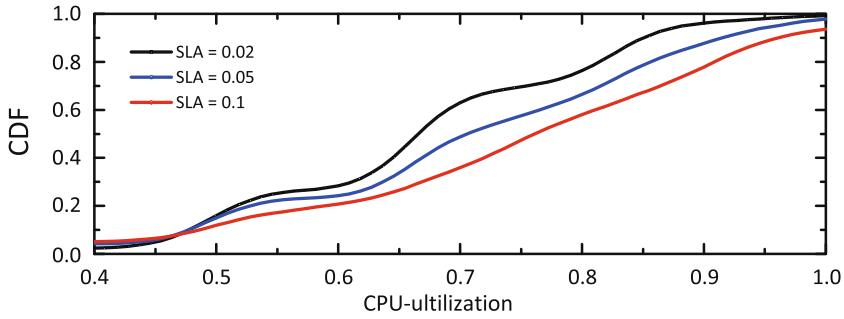

Fig. 6 The CDF of CPU-Utilization by correlative-aware FF algorithm under different user-defined SLAs

- Mutation Operation. Two or three PMs are selected and the minimum bin slack algorithm will be used to generate a better solution.

Constant Variance (CV) This algorithm predicts the future demand of VMs while not taking correlations between VMs into consideration [28].

3.4 Evaluation

The resource demand (utilization) data is generated by the method in [2]. We put 384 VMs on 128 PMs, and set the average resource demand of VMs belongs to $(0.165+\Delta)$, Δ obeys a normal distribution with an average of 0 and a variance of 0.04. We first generate 384 resource demand traces with different mean and variation. Each trace contains a list of 400 historical resource demand data (400 time slots). We will use the first 100 data to train the neural network model and the remaining data to compare our correlation-aware placement algorithms with previous proposed algorithms. We normalize the capacity of a PM as 100%. In Generic Algorithm, we crossover to 1000 generations and mutate 4 PMs every generation.

As shown in Table 4, Figs. 6 and 7, when the user-defined SLA becomes larger (2%-10%), more PMs achieve higher average resource utilization (about 10%), but the overload risk increases. There is a trade-off between resource utilization and user-defined SLA guarantee, and we should think twice before we make the decision under different scenarios.

As shown in Fig. 8, the resource utilizations of PMs are different under the four algorithms with user-defined SLA 5%. The parameter adjustment of Generic Algorithm is a tricky problem. The results of GA seem to the worst when we crossover to 1000 generations and mutate 4 PMs every generation. The constant variance (CV) algorithm makes an assumption that the variance of VM i is constant, which is not feasible in the real DC. Correlation-aware first-fit and best-fit algorithms outperforms the CV algorithm. When the overload ratios are almost the same, more PMs achieve higher resource utilizations and the total number of used PMs is lower by FF and BF algorithms. In spite of almost the same resource utilizations of PMs, the best-fit algorithm costs more than the first-fit algorithm due to searching the PM that can accommodate a VM with minimum free capacity at each time. The best outputs of the five algorithms is correlation-aware MBS algorithm, which uses the minimum number of PMs with maximum resource utilization, and the overload ratio is the nearest to the user-defined SLAs. For example, about 50% of PMs achieve 80% or more resource utilization. However, the cost of MBS algorithm is larger than FF and BF algorithms. For each PM, MBS algorithm

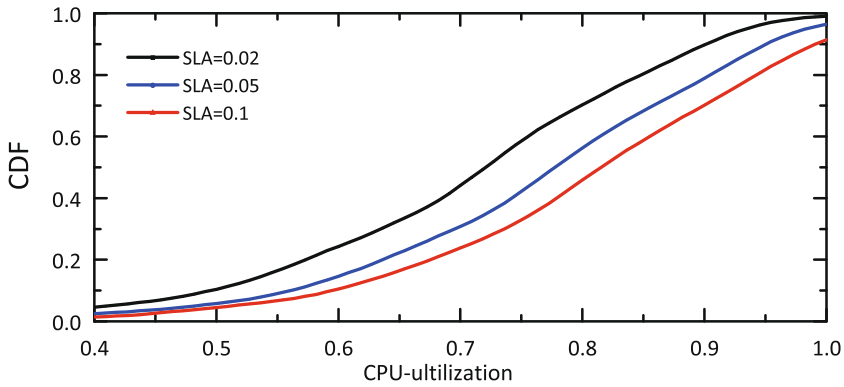

Fig. 7 The CDF of CPU-Utilization by correlative-aware MBS algorithm with different user-defined SLAs

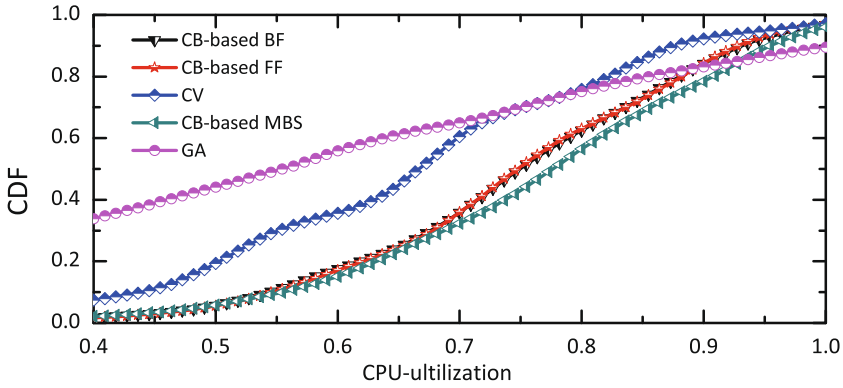

Fig. 8 Resource utilizations by different algorithms under user-defined SLA 5%

needs to search the subset of VMs to minimize the PM's slack.

4 Conclusion

In this paper, we proposed a correlation-aware virtual machine placement system that effectively places virtual machines on physical machines. First, we employ Neural Networks model to predict the resource utilization trend according to the historical resource utilization data. Second, we presented three correlation-aware VM placement algorithms to enhance resource utilization while meeting the user-defined service level agreements. The simulation results show that the efficiency of our virtual machine placement scheme outperforms the previous work by about 15%-30%.

Acknowledgements This work has been supported in part by Program of International S&T Cooperation (2016YFE0100300), China 973 project (2014CB340303), the National Natural Science Foundation of China (Grant number 61472252, 61672353, 61672349), and CCF-Tencent Open Research Fund. The authors also would like to thank Wei Wei for his contribution on the early versions of this paper.

References

1. Al-Fares M, Loukissas A, Vahdat A (2008) A scalable, commodity data center network architecture. Acm Sigcomm Comput Commun Rev 38(4):63–74
2. Ajiro Y, Tanaka A (2007) Improving packing algorithms for server consolidation. In: International CMG Conference, vol 253
3. Buyya R, Yeo CS, Venugopal S, Broberg J, Brandic I (2009) Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. Future Generation Comput Syst 25(6):599–616
4. Bobroff N, Kochut A, Beaty K (2007) Dynamic placement of virtual machines for managing sla violations. In: IFIP/IEEE International Symposium on Integrated Network Management (IM), pp 119–128
5. Cao B, Gao X, Chen G, Jin Y (2014) NICE: Network-Aware VM consolidation scheme for energy conservation in data centers. In: IEEE International Conference on Parallel and Distributed Systems (ICPADS), pp 166–173
6. Chen K, Singlay A, Singhz A, Ramachandranz K, Xuz L, Zhangz Y et al (2014) OSA: An optical switching architecture for data center networks with unprecedented flexibility. IEEE/ACM Trans Netw 22(2):498–511
7. Chen T, Gao X, Chen G (2016) The features, hardware, and architectures of data center networks: a survey. J Parallel Distributed Comput 96:45–74
8. Chen T, Zhu Y, Gao X, Kong L, Chen G, Wang Y (2016) Correlation-aware virtual machine placement in data center networks. In: 7th EAI International Conference on Cloud Computing (Cloudcomp), pp 1–10
9. Clark C, Fraser K, Hand S, Hansen JG, Jul E, Limpach C, Warfield A (2005) Live migration of virtual machines. In: USENIX Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation (NSDI), pp 273–286
10. Ghorbani S, Schlesinger C, Monaco M, Keller E, Caesar M, Rexford J, Walker D (2014) Transparent, live migration of a software-defined network. In: ACM Symposium on Cloud Computing (SOCC), pp 1–14
11. Gong Z, Gu X, Wilkes J (2010) Press: predictive elastic resource scaling for cloud systems. In: International Conference on Network and Service Management (CNSM), pp 9–16
12. Greenberg A, Hamilton JR, Jain N, Kandula S, Kim C, Lahiri P et al (2009) Vl2: a scalable and flexible data center network. Acm Sigcomm Comput Commun Rev 39(4):51–62
13. Guo C, Wu H, Tan K, Shi L, Zhang Y, Lu S (2008) Dcell: a scalable and fault-tolerant network structure for data centers. Acm Sigcomm Comput Commun Rev 38(4):75–86
14. Guo C, Lu G, Li D, Wu H, Zhang X, Shi Y et al (2009) Bcube: a high performance, server-centric network architecture for modular data centers. Acm Sigcomm Comput Commun Rev 39(4):63–74
15. Gupta JND, Ho JC (1999) A new heuristic algorithm for the one-dimensional bin-packing problem. Prod Plan Control 10(6):598–603
16. Han Z, Tan H, Chen G, Wang R, Chen Y, Lau F (2016) Dynamic virtual machine management via approximate markov decision process. In: IEEE International Conference on Computer Communications (INFOCOM), pp 1–9
17. Iima H, Yakawa T (2003) A new design of genetic algorithm for bin packing. Congress Evol Comput 2:1044–1049
18. Kai H, Bai X, Shi Y, Li M (2016) Cloud performance modeling with benchmark evaluation of elastic scaling strategies. IEEE Trans Parallel Distributed Syst 27(1):130–143
19. Kim J, Ruggiero M, Atienza D, Lederberger M (2013) Correlation-aware virtual machine allocation for energy-efficient datacenters. In: Proceedings of the Conference on Design, Automation and Test in Europe, pp 1345–1350

20. Khalilzad N, Faragardi HR, Nolte T (2015) Towards energy-aware placement of real-time virtual machines in a cloud data center. In: IEEE High Performance Computing and Communications (HPCC), pp 1657–1662
21. Lin H, Qi X, Yang S, Midkiff S (2015) Workload-driven VM Consolidation in cloud data centers. In: Parallel and Distributed Processing Symposium (IPDPS), pp 207–216
22. Meng X, Isci C, Kephart J, Zhang L, Bouillet E, Pendarakis D (2010) Efficient resource provisioning in compute clouds via vm multiplexing. In: International Conference on Autonomic Computing, pp 11–20
23. Nelson M, Lim BH, Hutchins G (2005) Fast transparent migration for virtual machines. In: Usenix Technical Conference, pp 391–394
24. Niu D, Feng C, Li B (2012) Pricing cloud bandwidth reservations under demand uncertainty. In: ACM Sigmetrics/performance Joint International Conference on Measurement and Modeling of Computer Systems, pp 151–162
25. Qiu C, Shen H, Chen L (2016) Probabilistic demand allocation for cloud service brokerage. In: IEEE International Conference on Computer Communications (INFOCOM), pp 1–9
26. Song W, Xiao Z, Chen Q, Luo H (2014) Adaptive resource provisioning for the cloud using online bin packing. IEEE Trans Comput 63(11):2647–2660
27. Wang S, Liu Z, Zheng Z, Sun Q, Yang F (2013) Particle swarm optimization for energy-aware virtual machine placement optimization in virtualized data centers. In: Parallel and Distributed Systems (ICPADS), pp 102–109
28. Wei W, Wei X, Chen T, Gao X, Chen G (2013) Dynamic correlative VM placement for quality-assured cloud service. In: IEEE International Conference on Communications (ICC), pp 2573–2577
29. Wood T, Shenoy P, Venkataramani A, Yousif M (2007) Black-box and gray-box strategies for virtual machine migration. In: Proceedings of the 4th USENIX conference on Networked systems design & implementation (NSDI). USENIX Association, pp 17–17
30. Wood T, Ramakrishnan KK, Shenoy P, Van der Merwe J, Hwang J, Liu G, Chaufournier L (2015) Cloudnet: dynamic pooling of cloud resources by live WAN migration of virtual machines. IEEE/ACM Transactions on Networking (TON) 23(5):1568–1583
31. Xu F, Liu F, Liu L, Jin H, Li B, Li B (2014) iAware: making live migration of virtual machines interference-aware in the cloud. IEEE Transactions on Computers (TOC) 63(12):3012–3025
32. Ye K, Jiang X, Huang D, Chen J, Wang B (2011) Live migration of multiple virtual machines with resource reservation in cloud computing environments. In: IEEE International Conference on Cloud Computing (CLOUD), pp 267–274
33. Yu L, Cai Z (2016) Dynamic scaling of virtual clusters with bandwidth guarantee in cloud datacenters. In: IEEE International Conference on Computer Communications (INFOCOM), pp 1–9
34. Yu L, Chen L, Cai Z, Shen H, Liang L, Pan Y (2016) Stochastic load balancing for virtual resource management in datacenters. IEEE Trans Cloud Comput:1–14. https://doi.org/10.1109/TCC.2016.2525984
35. Yu L, Shen H, Karan S, Ye L, Cai Z (2017) CoRE: cooperative end-to-end traffic redundancy elimination for reducing cloud bandwidth cost. IEEE Trans Parallel Distributed Syst 28(2):446–461
36. Zhou X, Zhang Z, Zhu Y, Li Y, Kumar S, Vahdat A et al (2012) Mirror mirror on the ceiling: flexible wireless links for data centers. ACM SIGCOMM Comput Commun Rev 42(4):443–454