IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON COLLABORATION FOR INTERNET OF THINGS

Received November 21, 2018, accepted November 28, 2018, date of publication November 30, 2018,
date of current version December 31, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2884251

Topology-Aware Resource Allocation for IoT Services in Clouds

XIN LI[1,2], (Member, IEEE), ZHEN LIAN[1], XIAOLIN QIN[1], AND JIE WU[3], (Fellow, IEEE)

[1]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China
[2]Information Technology Research Base of Civil Aviation Administration of China, Civil Aviation University of China, Tianjin 300300, China
[3]Center for Networked Computing, Temple University, Philadelphia PA 19122, USA

Corresponding author: Xin Li (lics@nuaa.edu.cn)

This work was supported in part by the National Key R&D Program of China under Grant 2018YFB1003900, in part by the National Natural Science Foundation of China under Grant 61802182, in part by the Jiangsu Natural Science Foundation under Grant BK20160813, in part by the Open Project Foundation of Information Technology Research Base of Civil Aviation Administration of China under Grant CAAC-ITRB-201706, and in part by the Collaborative Innovation Center of Novel Software Technology and Industrialization.

ABSTRACT With the development of the Internet of Things (IoT), the cloud data centers have already been an important foundation to support IoT data analysis and data-driven IoT services. For the data-driven services provision, cloud resources are necessary for the service components in the form of virtual machines (VMs). At the same time, there is a frequent data transmission among the service components (or VMs). Hence, to reduce the IoT services' response time, it is critical to improve the network issue and avoid network bottleneck during resource allocation. In this paper, we investigate the VM placement problem for balanced network utilization by avoiding network congestion. We first use the resource topology model to represent user requests and formulate the problem formally. We prove that the problem is NP-hard and present a heuristic algorithm based on the resource topologies. The core idea is to analyze the global and required resource topologies and place the required VMs into multiple servers with lower communication cost. We conduct extensive simulations, and the simulation results show that our algorithms have significant performance improvement on reducing network occupation and IoT service delay compared to the best-fit strategy and divide-and-conquer strategy.

INDEX TERMS Cloud data center, graph theory, IoT service, network optimization, virtual machine placement.

I. INTRODUCTION

Internet of Things (IoT) has created many exciting applications/systems, e.g. smart cities [1] and vehicular social network [2], [3]. These IoT systems generate big volume of data, and there is a strong need to conduct analysis on the big data [4]–[6] to support various data-driven services. Cloud data centers are the necessary platform to conduct data analysis and host various IoT services. For the IoT service provision, the service response time is critical to guarantee QoS (Quality of Service). In fact, the cloud resource allocation affects the service response time directly, not only the computing resource, but also the network resource. This is because there are always data transmissions between the service components.

Resource allocation is the primary issue in cloud data centers. For the IoT applications or services, there are always multiple service components or microservices [7], who actually occupy the resources in the form of virtual machines (VMs). Hence, VM placement is the concrete problem of resource allocation to satisfy the service resource requirements. According to the analysis of service provision, the VM placement should take both computing resource and network into account. However, it is still a challenging problem for various reasons [8], including the scalability issue, heterogeneous resources, workload variation, multiple optimization goals, even the network security problem [9]. These all make the VM placement (VMP) problem difficult.

Energy or cost reduction is the most important objective for the VMP problem in previous works. It is also known as power-based VMP, which presents a VM-PM (physical machine) mapping algorithm, allows a system to be energy-efficient and procures the utmost resource utilization [10], [11]. The power-based VMP algorithms mainly take the cost caused by physical resources into account. However, the Service-Level Agreement (SLA) violations are ignored. For example, it always take either the cost caused

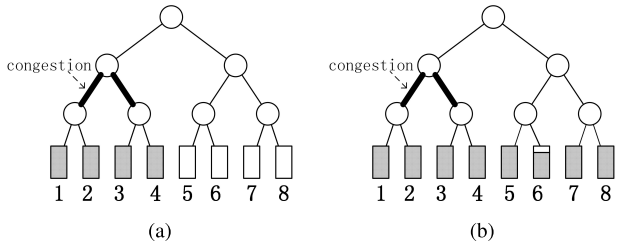


FIGURE 1. VMP with different goals. (a) minimize PM cost. (b) minimize network cost.

by PMs [12], [13] or the cost caused by network [14], [15] into account. But, the cost efficient approaches may lead to network congestion, which may deteriorate the data center performance and obviously enlarges the hosted IoT service delay. Such as in the literature [16], the authors summarizes the greenness of social sensor cloud but propose the outlook about social sensor cloud from the perspective of service.

To explain the above problem more clearly, there is an example in Fig. 1, a typical three-layer tree-like network topology. The rectangles represent PMs and the circles represent switches or routers. We assume that the resource requirements of the VM are the same and are represented by slots, including CPU, memory, disk, etc [17] and the PM has 8 slots. The cloud receives several requests R_1 at the same time and each request requires some VMs to complete the task. The numbers of VMs required are 4, 5, 5, 5, 6, 7. The VMs belonging to the same request need to work together to complete a task, so there is data communication between the VMs within the same request, but no traffic between any two different requests. We can easily get the VM placement solution while considering the cost caused by PMs, as shown in Fig. 1(a) and we know that there are only half of PMs are used, i.e. PM1, PM2, PM3, and PM4. In this solution, some requests are split into multiple parts and there are communication cost between different PMs, which may cause network congestion and performance reduction, as shown the bold line in Fig. 1(b). Though many previous works also consider the network cost and optimize the network issue by minimizing the overall network cost. For example, if the cloud receives two sets of requests at the same time R_1 and R_2, where R_2 contains 5 requests with numbers of required VMs are 3, 6, 7, 7, 8. For the same configuration, we can achieve the VM placement solution to minimize network cost, as shown in Fig. 1(b). The data communication may still occur within part of the network and this also may lead to network congestion, though it ensures the total network cost is minimal. In summary, most of the previous VM placement algorithms may cause network congestion, as the bold line shown in Fig. 1, if they take energy or the network cost as the major concern. Hence, for the IoT interactive services, these resource allocation algorithms are improper to utilize cloud resource. We need further take into account the network congestion during resource allocation.

In this paper, we eliminate the network congestion issue by minimizing the maximal link utilization and investigate the VMP problem for network congestion and bottleneck avoidance in cloud data center. We represent the user requirement by resource topology and formulate the problem, which is proved to be NP-hard. Then, we present our algorithm to make a simple optimization, based on topology analysis and graph theory. The core idea is to analyze the required VM resource topology and global useable resource topology, and split the VMs into multiple parts with fewer data communication. This is effective to utilize the network resource in a balanced manner. Compared to the best-fit and divide-and-conquer policies, the simulation results show that our algorithm has significant performance improvement in network utilization. Generally, our contributions can be summarized as follows.

(1) We formalize the VMP problem with the goal to achieve minimized maximal link utilization in data centers, and we prove that the problem is NP-hard. We mainly take the network issue into account to avoid network bottleneck and congestion. It is critical for the IoT services since data communication is necessary and frequent.

(2) We introduce the topology network to model the resource requirements and present a graph theory based heuristic algorithm, which is an improved version from our previous work [18]. We first analyze the global usable resources and resource requirements from users, and split the required resources (VMs) into multiple groups. But when the physical resources are insufficient and there is no bridge in the request resource topology, we will split the topology into two groups based on the minimal edge cut sets of resource topology.

(3) We conduct extensive simulations to evaluate the improved algorithm. The results show that the improved algorithm has better performance than our previous algorithm, which has significant performance improvement compared to the best-fit and divide-and-conquer algorithms.

The rest of the paper is organized as follows. We present the related work in Section II. The problem statement is given in Section III. We propose the improved algorithm in Section IV, and evaluate its performance in Section V. Finally, we make a conclusion in Section VI.

II. RELATED WORK

VM placement is the primary issue to achieve resource allocation is virtualization-based cloud data centers. The literatures have discussed the placement problem in various aspects, including resource utilization [19], [20], energy and cost [12], [13], [21], [22], network and scalability [14], [15], network overhead [23]–[25], network performance [26]–[29], and network congestion [30]–[34]. There are some other works, such as VM interference [35], [36], performance [37], VM migration [38], and cloudlet [39].

Li *et al.* [19] modeled the VMP problem as the multi-dimensional space partition considering the usage of multi-dimensional resources, such as CPU, memory, disk and etc. They tried to save energy by improving the utilization of resources. In the paper [20], they put forward to a

correlation-aware VMP that effectively deploys VMs on PMs while meeting the user-defined service level agreements (SLAs).

The authors model the VMP problem as the bin-packing problem and propose a VM placement and migration algorithm based on best-fit-decreasing for saving energy in [12]. But they cannot ensure that the user-defined SLAs, such as network congestion. The literature [13] aims to minimize the energy cost with considering constraints of PM resources and network bandwidth. the authors raise a solution based on the ant colony optimization for energy saving. Ahvar *et al.* [21] think out the energy saving problem as well as the emission of carbon in the content of geographic distribution of data centers. They propose an algorithm by combining prediction-based A* algorithm with Fuzzy Sets technique. Deng *et al.* [22] not only consider balancing the tradeoff between SLAs required by tenants and energy costs consumed by PMs, but also the lifetime and reliability of servers that are impacted by repeated on-off thermal cycles, wear-and-tear and temperature rise. They put forward a reliability-aware server consolidation to balance the multi-objective. Furthermore, the authors raise an algorithm based on the binary search and take the PM and network costs into account in [17].

Apart from energy saving, there are some works specifically minimizing network communication consumption in [14] and [15]. They try to deploy the VMs that need communicate with each other on the same PM or frame, which means the communication traffic flow among VMs through as few routers as possible. Meng *et al.* [14] try to deploy the VMs that need to communicate with each other on the only one PM. This scheme may achieve better results in some cases, but they just consider the overall network overhead, ignoring the utilization rate of network bandwidth and the solution easily generates hot spots even network congestion in the lower links.

Besides, there are some works notice the QoS while deploying the VMs. The works of literature [23], [24] both study the VMP problem considering the bandwidth required by VMs and aim to guarantee the QoS. Wang *et al.* [23] propose a solution to allocate computing and network resources for guaranteeing the QoS and balancing the resource utilization of PMs and bandwidth. Similarly, the authors give a solution which takes advantage of the one-step-ahead information to allocate bandwidth to VMs that hosting communication-intensive applications in [24]. Moreover, they also give a VM migration algorithm to adjust the bandwidth allocated when the network demands have variation, which aims to improve applications performance and reduce the overall traffic.

The authors raise a service-oriented architecture for VMP problem in [25]. They propose an algorithm based on integer linear programming for minimizing the communication time, that is to say, they need to allocate bandwidth as large as possible for each VM. Al-Fares *et al.* [26] improve the routing algorithm and apply it to the dynamic flow scheduling system of multiple tree topologies, which can adjust the uniform distribution of the flow on the network link, achieve traffic load balancing and improve the switch utilization so as to avoid the congestion of the data center link. Shrivastav *et al.* [27] use VMP to optimize network performance or end-to-end latency. It is recommended that VM migration is performed on overloaded PMs to balance the workload with the primary goal of eliminating overloaded PMs while reducing network congestion caused by migration traffic. Biran *et al.* [28] recommend optimizing network performance not only to meet predictable traffic communications needs but also to accommodate time-varying VM placement. However, neither of these studies considered the limitations of network link capacity and the optimization of maximal link utilization. In order to lower the communication latency, some works convert the VMP problem to the application placement in the mobile cloud network, such as the literature [29].

Silva and Fonseca [30] propose an algorithm for avoiding network congestion with considering the energy consumption of servers and switches and the basic idea of the algorithm is to occupy small resource of data center network. In addition, the authors not only give a solution for optimizing VMP but also a scheme for route selecting in [31] for the more connectivity and path diversity network architecture. The paper [32] takes advantage of the communication locality for balancing the communication traffic in data centers. But they just think out the upper links, ignore the lower links such as the links between physical servers and Top of Rack (ToR). Yan *et al.* [33] make use of the extended Hose model to deploy VMs. They put forward to a two-step solution for the problem of maximal link utilization, first propose a router assignment algorithm to balance the network bandwidth utilization, then adopt a heuristic algorithm to deploy VMs for eliminating network congestion. Son and Buyya [34] propose a priority-aware VM placement algorithm considering both host and network resources for reducing the chance of network congestion.

In our previous work, we propose a solution based on the bridge of graph theory in [18]. Experiments show that the scheme of VM placement can achieve an obvious effect in reducing link utilization. In this paper, we improve this method and conduct more simulations.

III. PROBLEM STATEMENT

In this section, we formalize the VM placement problem for minimizing maximal link utilization (MLU) and analyze its hardness.

A. PROBLEM DESCRIPTION

For a given cloud data center with three-layer tree-like network architecture, it contains N uniform PMs. For each PM, it has c resource slots to host VMs. Tenants or users submit their resource requirements, i.e. networked VMs, to the cloud data center. The resource requirement could be modeled by a weight undirected graph, where the VMs are represented by vertices while using edges to represent the traffic between

VMs and the weight of edge indicates the traffic volume. Hence, we can use a matrix to represent the user requirement. Formally, we use r_i to indicate the i^{th} user request, i.e. the resource topology or the weight undirected graph.

For the cloud data center, it receives a set of user requirements R, where $R = \{r_i,\ 0 \le i \le n\}$. It needs to allocate the required resources for users based on their required resource topologies. For each user requirement, if the required VMs are placed on the same PM, it is known as perfect placement, which indicates that there is no extra communication cost. On the other hand, the VMs of the same request may be split into multiple parts, and each part is hosted on one PM. This will lead to traffic between PMs who host the VMs from the same request. In fact, the VMs are always be split into multiple parts due to the limited resource capacity for PM. The objective of the VM placement is to achieve minimized maximal link utilization (MLU), which could reduce network congestion and bottleneck.

To formally define the problem, we use $link(s, t)$ to represent the edge and its weight between node s and t, which could be PM or switch. The bandwidth of $link(s, t)$ is given by $b(s, t)$. For the communication traffic between VM_i and VM_j, the part hosted by link (s,t) is represented by $P^{(i,j)}_{(s,t)}$. For each link(s,t), we define its link utilization $U(s,t)$ as

$$U(s,t) = \sum_{(i,j)} P^{(i,j)}_{(s,t)} / b(s,t) \tag{1}$$

Hence, we can formalize the VMP problem for minimizing MLU as follows.

VM placement for minimizing MLU: For a given data center with uniform PMs, it accepts a set of resource requests $R=\{r_i|0\le i\le n\}$. Given a VM placement scheme such that the MLU is minimized. It can be formalized as

$$\begin{aligned} &min\ max\{U(s,t)\} \\ &s.t.\ P^{(i,j)}_{(s,t)} = P^{(i,j)}_{(t,s)} \\ &\quad \sum_{(i,j)} P^{(i,j)}_{(s,t)} \le b(s,t) \\ &\quad P^{(i,j)}_{(s,t)} \ge 0 \end{aligned} \tag{2}$$

where the $P^{(i,j)}_{(s,t)} = P^{(i,j)}_{(t,s)}$ means the flow of any two VMs through the same link is equal, $\sum_{(i,j)} P^{(i,j)}_{(s,t)} \le b(s,t)$ indicates the link capacity, and the traffic amount should be greater than 0, i.e. $P^{(i,j)}_{(s,t)} \ge 0$.

B. HARDNESS

Theorem 1: The VM placement for minimizing MLU problem is NP-hard.

Proof: We will prove the theorem by showing a special case is NP-hard. We construct the special case: There are two PMs in the data center, and the traffic flow is constant. We assume that the sum of required VMs equals to the capacity of the two PMs. Hence, the optimal solution is to divide the requests into two equal parts without splitting any request and place each part on one PM. In this case, there is no network cost, it achieves the minimized MLU. Next, we will show that minimizing MLU is NP-hard in this case.

First, it is easy to verify the feasibility of a given solution in polynomial time. Then, we show that the minimizing MLU problem can be reduced from the *subset-sum* problem. The *subset-sum* problem can be formalized as follows: given a set of integers $A=\{A_1,A_2,...,A_n\}$, determine whether there is a subset A^* of A, such that $\sum_{A_i\in S^*} A_i = \sum_{A_i\in S} A_i/2$.

Therefore, we can construct the problem of minimizing MLU. Let there are n requests $\{r_1,r_2,...,r_n\}$ and two PMs $\{P_1,P_2\}$. For the requests, let $r_i = A_i$ for each request, and the capacity of each PM equal to $\sum_{i=1}^{n} r_i/2 = \sum_{i=1}^{n} A_i/2$. Here, if there exist a subset $A^* \subset A$, such that $\sum_{A_i\in A^*} A_i = \sum_{A_i\in A} A_i/2$, we will place the VMs in A^* on PM P_1, and place others on P_2. There is no network cost with this placement since none of the requests is placed with partition. On the other hand, if there is a VM placement with the network cost equals to 0, the requests in one PM could be a feasible solution of A^* for the sub-set problem.

Since the typical sub-set problem is known as NP-hard, we conclude that the VM placement for minimizing MLU problem is NP-hard. ■

Algorithm 1 Bridge-based Placement $BBP(N, c, n, r)$

Require: N: number of PMs; c: capacity of PMs; n: number of requests; r: weighted undirected graph.

```
all requests in descending order according to the number of VMs;
for i = 0 → n − 1 do
  if ∃ c ≥ r_i then
    perfect placement;
  else
    if find bridges(r_i) then
      remove the bridges and sort the subgraphs in ascending order by number of nodes;
      if ∃ c ≥ r_ij then
        allocate the r_ij to the best-fit PM;
      else
        strategy_placement(N,c,r_ij);
    else
      strategy_placement(N,c,r_i);
```

IV. TOPOLOGY-AWARE VM PLACEMENT

In the Section III, we give the description of VMP in detail, including the form of tenant request and the optimization goals. We let a PM can hold at least one VM, that is $\forall i,\ r_i<c$. Besides, we also take the utilization rate of network links and communication overhead into account. In this section, we put forward an algorithm that utilizes the bridge partitioning the request topology when the resource is insufficient. However, the performance of our algorithm will drop when there is no bridge in the resource topology. In order to improve this situation, we use the minimal edge cut set to replace the bridge to cut up the request.

Algorithm 2 Best-Fit Strategy $BFS(N, c, r_i)$

Require: N: number of PMs; c: capacity of PMs; r_i: weighted undirected graph.

1: find a node k with max traffic flow;
2: find a PM p with best-fit(r_i) remaining capacity c_{cap};
3: **for** $i = 0 \rightarrow C_{cap}$ **do**
4: allocate k and adjoining to it to p;
5: **if** ∃ nodes belonging to r_i are not allocated **then**
6: $BFS(N, c, r_i)$;

Algorithm 3 Divide-and-Conquer Strategy $DCS(N, p, c, r_i)$

Require: N: number of PMs; p: the current PM; c: capacity of PMs; r_i: weighted undirected graph.

1: find a node k with max traffic flow;
2: find a PM adjacent to p with capacity c_{cap} as new p;
3: **for** $i = 0 \rightarrow C_{cap}$ **do**
4: allocate k and adjoining to it to p;
5: **if** ∃ nodes belonging to r_i are not allocated **then**
6: $DCS(N, p, c, r_i)$;

We propose a solution based on the bridge of request topology for minimizing MLU in Algorithm 1. The basic idea is to conduct *perfect placements* as many as possible. When data center resources are scarce, we need to divide a request into several parts according to the bridges in the request resource topology and the request is divided into several sub-requests. These sub-requests are sorted in ascending order by the number of VMs required and we use best-fit strategy to deploy sub-requests. But the following will still occur: a sub-requests cannot be allocated to the PM perfectly due to the VMs required are not satisfied. We first choose the VM with the maximal bandwidth requirement and use best-fit strategy (BFS) or divide-and-conquer strategy (DCS) to deploy it and VMs which are communicating with it. Unlike previous work with the goal of minimizing network cost, our algorithm tries to make the network load balance so as to avoid network congestion.

In the Algorithm 1, we first sort the all requests in ascording order according to the VMs required. If there is a PM that can hold the current request r_i, the request can be placed perfectly *(line 3-4)*. This will ensure that all the requests can be placed perfectly when resources are sufficient, so there is no traffic in the data center. However, when the remaining resources are insufficient and the request cannot be placed perfectly, we will analyze the topology of the request, find all bridges of the request r_i and remove them *(line 6-7)*. For the split request, if there is a PM that can hold the sub-request r_{ij} and it will be placed perfectly *(line 8-9)*, the r_{ij} represents the j-th sub-request of r_i. However, if the remaining capacity of any PM is insufficient for r_{ij} or there is no bridge in request r_i, we use strategies placement *(line 10-13)*. We propose two different strategies: best-fit strategy (BFS) in Algorithm 2 and divide-and-conquer strategy (DCS) in Algorithm 3.

We make use of the best-fit strategy in the Algorithm 2 to solve the situation that the resource is insufficient as described at *line 10-13* in the Algorithm 1. We first find a VM with the maximal traffic *(line 1)* and search a best-fit PM *(line 2)*. We will deploy the VM and others adjoining to it on the PM until all the VMs are placed *(line 3-6)*. The best-fit strategy can improve resource utilization to a certain extent by taking advantage of the remaining capacity.

The divide-and-conquer strategy is elaborated in the Algorithm 3, we divide the network topology into several sub-trees from the core layer and the convergence layer. We will first search a PM in a sub-tree but if the sub-tree does not have enough resources to satisfy the request, the request will be deployed to another subtree that adjacent to the current. So the algorithm based on the divide-and-conquer strategy generates communication cost as low as possible. Like the Algorithm 2, we also find a VM with the maximal traffic *(line 1)* and search a PM adjacent to p *(line 2)*, if the remaining capacity of the current PM is greater than zero, then the target is the current PM. We will deploy the VM and others adjoining to it in the current sub-tree until all the VMs are placed *(line 3-6)*.

However, after extensive simulations are conducted, we find that when the physical resources are insufficient or the request resource topology does not have a bridge, the BBP has an obvious shortcoming that degenerates into best-fit or divide-and-conquer. And we find the divide-and-conquer strategy has the best performance. Therefore, we improve the flip algorithm BBP and use the minimal edge cut set instead of the bridges. We describe the edge-cut set based placement in Algorithm 4. First, we still descend all requests by the number of VMs required and implement as many *perfect placement* as possible *(line 1-4)*. But when no PM can contain the request, we find an edge-cut set of the request topology, which has the smallest weight, and remove it *(line 6-7)*, rather than finding the bridges. Then, the weighted undirected graph can be divided into only two parts and use best-fit strategy to allocate them *(line 8-9)*. Otherwise, we still adopt divide-and-conquer described in Algorithm 3 place them on PMs, respectively *(line 10-11)*.

For a better understanding, we give an example of the implementation process step by step about the Algorithm 1 with best-fit and divide-and-conquer strategies based on the three-layer tree-like network architecture as shown in Fig. 1. In order to better prove the effectiveness of the algorithm, we only use PM_1, PM_2, PM_3 and PM_4. There is a set of requests that need to be deployed and the numbers of VMs required are $R = \{7,6,5,5,5,4\}$. Besides, the requests topology is known as follows: the requests are sorted in descending order according to VMs required, so we ignore the resource topology of r_1, r_2, r_3 and r_4 because they can be placed perfectly without segmentation. r_5 has only one bridge and it can be divided into r_{51} with 1 VM and r_{52} with 4 VMs, r_6 does not have a bridge. The deployment scheme is shown in Fig. 2, and the algorithm is executed as follows:

(1) r_1, r_2, r_3, and r_4 are placed perfectly;

Algorithm 4 Edge-Cut Based Placement $ECBP(N, c, n, r)$

Require: N: number of PMs; c: capacity of PMs; n: number of requests; r: weighted undirected graph.

```
all requests in descending order according to the number of VMs;
for i = 0 → n − 1 do
    if ∃ c ≥ r_i then
        perfect placement;
    else
        if find the edge cut set which has the smallest weight(r_i) then
            remove the edge cut set and the r_i is divided into r_i1 and r_i2;
            if ∃ c_j ≥ r_ij then
                allocate the r_ij to the best-fit PM c_j;
            else
                DCS;
```

(2) r_5 is partitioned into two parts and r_{51} is deployed on PM_1 seemly.

(3) r_{52} is partitioned into r_{52-1} with 3 VMs and r_{52-2} with 1 VM, while r_{52-1} is deployed on PM_3. The Fig. 2(a) is outcome of the Algorithm 2: r_{52-2} is deployed on PM_2 and r_6 is partitioned into 1 VM and 3 VMs that are deployed on PM_2 and PM_4, respectively. The Fig. 2(b) is outcome of the Algorithm 3: r_{52-2} is deployed on PM_4 and r_6 is divided into two same parts that are deployed on PM_2 and PM_4.

For the improved edge-cut set based placement, for example, there is a request set with the numbers of VMs required are R={*7,7,6,6,6*} and the requests topology are described as follows: the requests are sorted in descending order according to the number of VMs required, so we ignore the resource topology of r_1, r_2, r_3, and r_4, because they can be deployed perfectly without segmentation. r_5 has no bridge, but it can be separated into two parts, r_{51}, r_{52} with 2 VMs and 4 VMs by the minimal edge cut set. However, the bridge-based placement Algorithm 1 cannot find the minimal edge-cut set, so r_5 cannot be split. The deployment scheme is shown in Fig. 3 and the Algorithm 4 is executed as follows:

(1) r_1, r_2, r_3, and r_4 are deployed perfectly;

(2) r_5 is partitioned into two parts: r_{51} and r_{52} with 2 VM and 4 VMs, respectively. In addition, r_{51} is placed on PM_3 suitably.

(3) Finding two edge cut sets of r_{52} so that it is divided into r_{52-1}, r_{52-2} and r_{52-3} with 2 VMs, 1 VM and 1 VM according to the remaining capacity of PMs, respectively. r_{52-1} is placed on PM_4, r_{52-2} is placed on PM_1 and r_{52-3} is placed on PM_2.

To research the performance of Algorithm 1 with different deployment strategies in theory, we classify the kinds of request resource topology into several different categories: without a bridge, only one bridge and more than one bridge. For the case of without bridge, we adopt the deployment of Algorithm 2 or Algorithm 3 to allocate the VMs required, so the effect is the same as best-fit or divide-and-conquer algorithm. If a request has one or more bridges and it is partitioned into two or more sub-requests. However, when there are not enough resources to satisfy sub-requests, we will adopt the best-fit or divide-and-conquer deployment described in the Algorithm 2 or Algorithm 3 to place them. Therefore, the effect of our scheme is similar to best-fit or divide-and-conquer placement. If these subgraphs can be placed perfectly, so there are at most only one bridge's cost, but best-fit or divide-and-conquer deployment will produce more communication traffic for the case of only one bridge. If the request has more than one bridge, we assume an extreme situation that all edges in the request are bridges. The effect of our algorithm will be not worse than the best-fit or divide-and-conquer strategy for the worst case. Hence, we infer that the Algorithm 1 based on the bridge is not worse than best-fit or divide-and-conquer placement. The edge-cut set based placement is an improvement of bridge-based placement; when there is no bridge in the request, we will find a minimal edge-cut set to replace it so as to reduce the overall traffic flow.

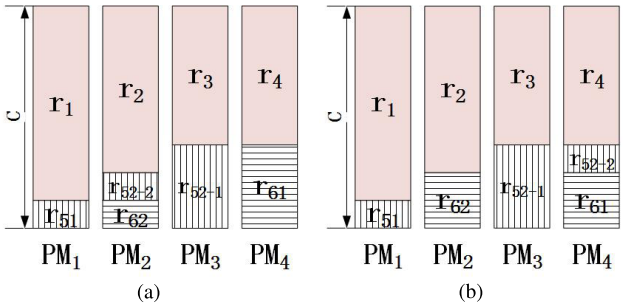


FIGURE 2. the result of different strategies. (a) best-fit strategy. (b) divide-and-conquer strategy.

About the time complexity of our algorithm, it can find all bridges within O(n+m) in the worst case, where the n represents the VMs required by a request and m is the number of communication relationships. So, we can deploy a request within O(n(n+m)). The edge-cut set based placement can find the minimal edge-cut set within $O(n^3)$ and deploy a request within $O(n^4)$. In short, the execution time of our algorithm is within acceptable limits.

V. EVALUATION

We describe our algorithm and the performance in theory detailed in the last section, we adopt best-fit and divide-and-conquer algorithm as deployment strategies. Therefore, best-fit and divide-and-conquer algorithms are baselines to evaluate our algorithms. We simulate five algorithms and compare with each other under different workloads. Our algorithms are proved to be effective and we demonstrate the improvement of our algorithm's performance along with the addition of load.

A. ALGORITHM DESCRIPTION

The basic idea of the best-fit based placement (BFBP) is all PMs are sorted in ascending order according to the remaining

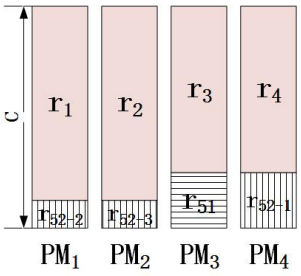


FIGURE 3. the result of minimal edge-cut set placement.

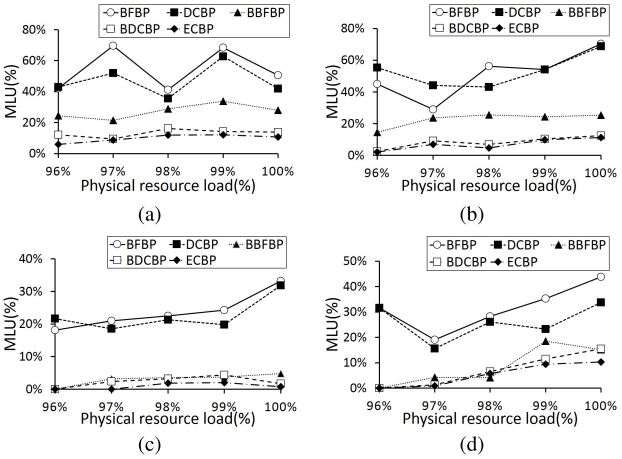


FIGURE 4. The MLU of five algorithms under different distributions of VMs and traffic with the data center workload from 96% ∼ 100%. (a) VMs required are uniform distribution with lower limit 2 and traffic is uniform distribution. (b) VMs required are uniform distribution with lower limit 2 and traffic is normal distribution. (c) VMs required are normal distribution with lower limit 2 and traffic is uniform distribution. (d) VMs required are normal distribution with lower limit 2 and traffic is normal distribution.

resources, to ensure that we can find the best fit PM for the request. However the best-fit placement may cause the fragmentation problem, because the remaining resource will be so small that can not contain any requests or sub-requests. Therefore, when the data center resource load is high and the resource is insufficient, requests may be partitioned into so many pieces to suit the remaining resource of PM that the utilization of physical links is high. However, the best-fit placement may improve resource utilization to a certain extent.

For the divide-and-conquer placement (DCBP) algorithm, we will divide the tree-like network topology of data centers into sub-trees from the core layer and convergence layer and deploy requests in every sub-tree. The request will be deployed to the neighboring when the remaining resources of a sub-tree communication but ignore the load of network bandwidth. This deployment scenario may lead to the *network hotspot* even the network congestion.

The above two deployment scenarios are opted as baselines to assess our algorithms. Based on the above algorithms, we combine two approaches: the bridge-and-best-fit based placement (BBFBP) is the combination of bridge-based placement 1 and best-fit strategy 2, while the bridge-and-divide-and-conquer based placement (BDCBP) consists of bridge-based placement 1 and divide-and-conquer strategy 3. In addition, The edge-cut based placement (ECBP) 4 has been introduced in Section IV.

B. SIMULATION SETTINGS

We adopt uniform distribution and normal distribution of VMs required by a request and the number of VMs required is less than the resources owned by a PM. This guarantees that there are some requests are deployed perfectly. Besides, the communication traffic flow size among VMs is also uniformly or normally distributed within the predetermined range.

For the three-layer tree-like network architecture, the root is a core router and there are two converged routers in the second layer, the bandwidth between root and its child nodes is *30000M*. Each converged router has three child node that is cabinet top switch and the bandwidth between the two layers is *3000M*. The cabinet top switch connects to all 20 PMs belong to the cabinet and each PM has enough resources to hold 20 VMs. So we let each request requires up to *20* VMs and at least *2* VMs. In addition, the bandwidth between each PM and the cabinet top switch is *300M*. The workload of the data center is scattered in 96% ∼ 100% with granularity is 1%. Due to the lower limit of the number of VMs required by the request, if the number of requests is not enough to occupy the resources, a large number of requests will be deployed perfectly. This situation results in the MUL is very small even zero. Therefore, we make the workload of data center is relatively high. We increase the lower limit to *10* of VMs required by the request in another simulation experiment and reduce the workload properly of data center, the workload of data center is scattered in 80% ∼ 100% with the granularity is 5%. The communication traffic size is uniformly or normally distributed with the lower limit is 0 and upper limit is $5M$. Besides, we let the second layer has *5* converged routers to expand the scale of experiments.

C. SIMULATION RESULTS

According to the above experimental environment, we conduct extensive simulations. There is a set of experimental results shown in Table 1 where VMs are uniformly distributed in 10∼20 and traffic is uniformly distributed in 0∼5. The content in the first column is resource load of data center. The second to the sixth column are the MLU of five algorithms. And the seventh column is effect of algorithm ECBP compared to BDCBP which is most outstanding in [18]. From the table, we can see the algorithm ECBP is the best solution and the MLU declines about 4% compared to BDCBP. All experimental results are given in form of figure as follows.

Fig. 4 shows a set of experimental results with the following settings: the number of VMs required by a request has the lower limits *2* and the workload of data center is from

TABLE 1. VMs are uniformly distributed in 10∼20, traffic is uniformly distributed.

MLU(%) / workload(%)	$BFBP$	$DCBP$	$BBFBP$	$BDCBP$	$ECBP$	$effect$
80	44.52	44.52	20.79	8.69	5.41	**3.28**
85	60.76	68.20	27.21	17.27	13.88	**3.39**
90	69.84	61.98	28.89	23.35	19.82	**3.53**
95	82.04	72.35	40.33	27.85	22.85	**5.00**
100	78.31	79.05	45.96	30.44	30.02	**0.42**

96% to 100%. The MLU in the Figs. 4(a) and 4(b) where the VMs required are uniform distribution is obviously larger than its in the Figs. 4(c) and 4(d) with normal distribution of VMs. Due to added requests with the number of VMs close to *2* or *20* in the Figs. 4(a) and 4(c). The communication traffic flow is uniformly distributed in Figs. 4(b) and 4(d) and normally distributed in Figs. 4(a) and 4(b). The BFBP algorithm almost has the largest link bandwidth utilization, especially when the VMs required are normal distribution and the DCBP algorithm is seemly like to the BFBP in the Figs. 4(a) and 4(b). We can infer that the BBFBP algorithm is more outstanding than the above two algorithms and the MLU declines around 30% in Figs. 4(a) and 4(b) where the VMs required are the uniform distribution. Besides, the MLU of BDCBP declines around 15% compared to BBFBP algorithm when the VMs required are the uniform distribution in Figs. 4(a) and 4(b). In addition, the performance of improved ECBP is best in the five deployment schemes and the MLU of ECBP algorithm drops around 3% ∼ 5%. The MLU is showing an uptrend along with the increase of data center workload, the performance of BBFBP and BDCBP are roughly the same and more excellent than BFBP and DCBP, moreover, the ECBP always has the smallest MLU. Above all, the simulation results can fully reflect the superiority of the bridge-based algorithm and the edge-cut-based algorithm in various cases.

The Fig. 5 is the results of five algorithms with the number of VMs required has a lower limit *10* and the workload of the data center is from 80% to 100%. In the Fig. 5(a), the VMs required and communication traffic are uniform distribution, the MLU of five deployment is increasing along with the growth of data center workloads. The ECBP has the most outstanding performance and the MLU of ECBP declines at least 45% and 40% compare to the BFBP and DCBP. Compared to the Fig. 5(a), the settings difference of Fig. 5(b) is that the communication traffic is normal distribution. With the communication traffic size increasing, the MLU of our algorithms is also becoming bigger, but our algorithms are still more effective than others and the MLU drops about 30%. The VMs required are normal distribution in Fig. 5(c) and Fig. 5(d), but the distribution of communication traffic is different. The results in the Fig. 5(c) are similar to the Fig. 5(a), but the performance improved at least 30%. The communication traffic is normal distribution in the Fig. 5(d) and the ECBP has the best performance with the MLU drops

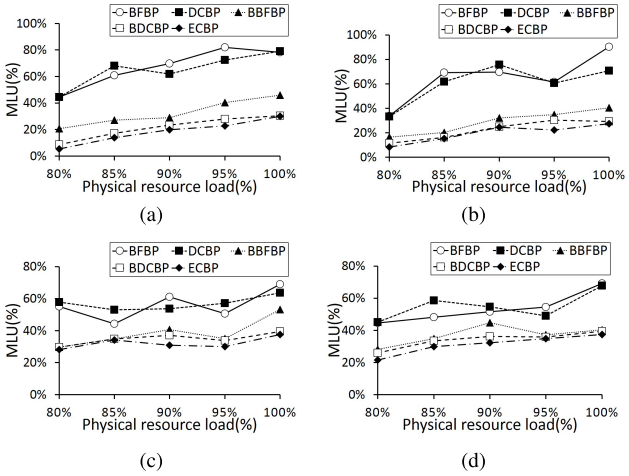


FIGURE 5. The MLU of five algorithms under different distributions of VMs and traffic with the data center workload from 80% ∼ 100%. (a) VMs required are uniform distribution with lower limit 10 and traffic is uniform distribution. (b) VMs required are uniform distribution with lower limit 10 and traffic is normal distribution. (c) VMs required are normal distribution with lower limit 10 and traffic is uniform distribution. (d) VMs required are normal distribution with lower limit 10 and traffic is normal distribution.

around 4% compared to the bridge based algorithms, and declines around 40% compared to the BFBP. Through the above sets of experiments we can infer that the influence of VM's distribution comparing the Figs. 5(a) and 5(c), or 5(b) and 5(d). Through the Figs. 5(a) and 5(b), or 5(c) and 5(d) we can conclude the influence of different communication traffic distribution on the MLU. In summary, the bridge based algorithms and edge-cut set based algorithm are all better than BFBP and DCBP, of which ECBP has the smallest MLU whatever the data center workloads.

In the Fig. 6, we let the second layer of the network architecture has 5 aggregate routers to increase the resources of data center and the other settings are same as Fig. 5, we also conducted a large number of simulation experiments. Experimental results show that the bridge-based algorithms are more outstanding the BFBP and DCBP in various situations and the ECBP is the optimal deployment scheme. For example, the bridge-based algorithms are strictly superior the BFBP and DCBP from the Figs. 6(a) and 6(b) and the performance of ECBP is further improved. Besides, from the Figs. 6(c) and 6(d), we can infer that the DCBP and BDCBP has a slight

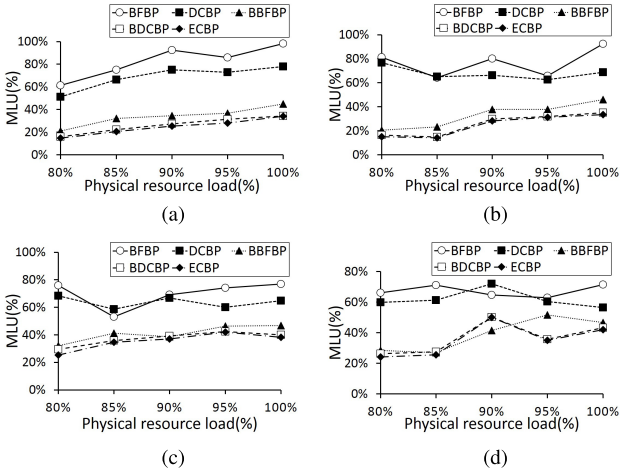


FIGURE 6. The MLU of five algorithms under different distributions of VMs and traffic with the data center workload from 80% ∼ 100%, the scale of data center is enlarged. (a) VMs required are uniform distribution with lower limit 10 and traffic is uniform distribution. (b) VMs required are uniform distribution with lower limit 10 and traffic is normal distribution. (c) VMs required are normal distribution with lower limit 10 and traffic is uniform distribution. (d) VMs required are normal distribution with lower limit 10 and traffic is normal distribution.

fluctuation, but bridge-based algorithms and ECBP still have a satisfactory result on the whole.

In short, the bridge-based algorithms and improved edge-cut set based have an outstanding and stable performance regardless of what the data center workload and scale are or what the distribution of VMs or traffic is.

VI. CONCLUSION

We study the VM placement problem for minimizing MLU in this paper. We formalize the problem and prove its hardness. To deal with the placement issue, we propose a graph theory based heuristic algorithm. The basic idea is to take into account the resource topologies of requests and employ various placement strategies under insufficient resources. Furthermore, we improve the algorithm against the situation that physical resources are insufficient or the request topology has no bridge. We conduct extensive simulations to evaluate our algorithms, and the results show that the proposed idea has significant improvement than the classical best-fit and divide-and-conquer manner.

REFERENCES

[1] C. Zhu, H. Zhou, V. C. M. Leung, K. Wang, Y. Zhang, and L. T. Yang, "Toward big data in green city," *IEEE Commun. Mag.*, vol. 55, no. 11, pp. 14–18, Nov. 2017.

[2] C.-M. Huang, Y.-F. Chen, S. Xu, and H. Zhou, "The vehicular social network (VSN)-based sharing of downloaded geo data using the credit-based clustering scheme," *IEEE Access*, vol. 6, pp. 58254–58271, 2018.

[3] H. Zhou, H. Wang, X. Chen, X. Li, and S. Xu, "Data offloading techniques through vehicular ad hoc networks: A survey," *IEEE Access*, vol. 6, pp. 65250–65259, 2018.

[4] H. Zhou, S. Xu, D. Ren, C. Huang, and H. Zhang, "Analysis of event-driven warning message propagation in vehicular ad hoc networks," *Ad Hoc Netw.*, vol. 55, pp. 87–96, Feb. 2017.

[5] W. Chang and J. Wu, "Privacy-preserved data publishing of evolving online social networks," *J. Inf. Privacy Secur.*, vol. 12, no. 1, pp. 14–31, 2016.

[6] W. Jiang, J. Wu, G. Wang, and H. Zheng, "Forming opinions via trusted friends: Time-evolving rating prediction using fluid dynamics," *IEEE Trans. Comput.*, vol. 65, no. 4, pp. 1211–1224, Apr. 2016.

[7] Y. Niu, F. Liu, and Z. Li, "Load balancing across microservices," in *Proc. IEEE INFOCOM*, Apr. 2018, pp. 198–206.

[8] F. Xu, F. Liu, H. Jin, and A. V. Vasilakos, "Managing performance overhead of virtual machines in cloud computing: A survey, state of the art, and future directions," *Proc. IEEE*, vol. 102, no. 1, pp. 11–31, Jan. 2014.

[9] H. Zhang, Y. Qi, H. Zhou, J. Zhang, and J. Sun, "Testing and defending methods against DOS attack in state estimation," *Asian J. Control*, vol. 19, no. 3, pp. 1295–1305, 2017.

[10] F. Liu, Z. Zhou, H. Jin, B. Li, B. Li, and H. Jiang, "On arbitrating the power-performance tradeoff in SaaS clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 10, pp. 2648–2658, Oct. 2014.

[11] B. Gohil, S. Shah, Y. Golechha, and D. Patel, "A comparative analysis of virtual machine placement techniques in the cloud environment," *Int. J. Comput. Appl.*, vol. 156, no. 14, pp. 12–18, 2016.

[12] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Generat. Comput. Syst.*, vol. 28, no. 5, pp. 755–768, 2012.

[13] C. Gao, H. Wang, L. Zhai, Y. Gao, and S. Yi, "An energy-aware ant colony algorithm for network-aware virtual machine placement in cloud computing," in *Proc. IEEE 22nd Int. Conf. Parallel Distrib. Syst. (ICPADS)*, Dec. 2016, pp. 669–676.

[14] X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.

[15] K. You, B. Tang, and F. Ding, "Near-optimal virtual machine placement with product traffic pattern in data centers," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2013, pp. 3705–3709.

[16] C. Zhu, V. C. M. Leung, J. J. P. C. Rodrigues, L. Shu, L. Wang, and H. Zhou, "Social sensor cloud: Framework, greenness, issues, and outlook," *IEEE Netw.*, vol. 32, no. 5, pp. 100–105, Sep./Oct. 2018.

[17] X. Li, J. Wu, S. Tang, and S. Lu, "Let's stay together: Towards traffic aware virtual machine placement in data centers," in *Proc. IEEE INFOCOM*, Apr./May 2014, pp. 1842–1850.

[18] Z. Lian, X. Li, and X. Qin, "Topology-aware VM placement for network optimization in cloud data centers," in *Proc. IEEE Int. Symp. Parallel Distrib. Process. Appl. (ISPA)*, Dec. 2017, pp. 558–565.

[19] X. Li, Z. Qian, S. Lu, and J. Wu, "Energy efficient virtual machine placement algorithm with balanced and improved resource utilization in a data center," *Math. Comput. Model.*, vol. 58, no. 5, pp. 1222–1235, 2013.

[20] T. Chen, Y. Zhu, X. Gao, L. Kong, G. Chen, and Y. Wang, "Improving resource utilization via virtual machine placement in data center networks," *Mobile Netw. Appl.*, vol. 23, no. 2, pp. 227–238, 2018.

[21] E. Ahvar, S. Ahvar, Z. A. Mann, N. Crespi, J. Garcia-Alfaro, and R. Glitho, "CACEV: A cost and carbon emission-efficient virtual machine placement method for green distributed clouds," in *Proc. IEEE Int. Conf. Services Comput. (SCC)*, Jun./Jul. 2016, pp. 275–282.

[22] W. Deng, F. Liu, H. Jin, X. Liao, H. Liu, and L. Chen, "Lifetime or energy: Consolidating servers with reliability control in virtualized cloud datacenters," in *Proc. IEEE Int. Conf. Cloud Comput. Technol. Sci. (CloudCom)*, Dec. 2012, pp. 18–25.

[23] R. Wang, J. A. Wickboldt, R. P. Esteves, L. Shi, B. Jennings, and L. Z. Granville, "Using empirical estimates of effective bandwidth in network-aware placement of virtual machines in datacenters," *IEEE Trans. Netw. Service Manage.*, vol. 13, no. 2, pp. 267–280, Jun. 2016.

[24] D. Shen, J. Luo, F. Dong, and J. Zhang, "AppBag: Application-aware bandwidth allocation for virtual machines in cloud environment," in *Proc. 45th IEEE Int. Conf. Parallel Process. (ICPP)*, Aug. 2016, pp. 21–30.

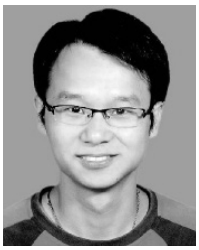
[25] F.-H. Tseng, Y.-M. Jheng, L.-D. Chou, H.-C. Chao, and V. C. M. Leung, "Link-aware virtual machine placement for cloud services based on service-oriented architecture," *IEEE Trans. Cloud Comput.*, to be published, doi: 10.1109/TCC.2017.2662226.

[26] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, and A. Vahdat, "Hedera: Dynamic flow scheduling for data center networks," in *Proc. USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2010, p. 19.

[27] V. Shrivastava, P. Zerfos, K.-W. Lee, H. Jamjoom, Y.-H. Liu, and S. Banerjee, "Application-aware virtual machine migration in data centers," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 66–70.

[28] O. Biran *et al.*, "A stable network-aware VM placement for cloud systems," in *Proc. 12th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput.*, May 2012, pp. 498–506.

[29] W. Tärneberg *et al.*, "Dynamic application placement in the mobile cloud network," *Future Gener. Comput. Syst.*, vol. 70, pp. 163–177, May 2017.
[30] R. A. C. da Silva and N. L. S. da Fonseca, "Topology-aware virtual machine placement in data centers," *J. Grid Comput.*, vol. 14, no. 1, pp. 75–90, 2016.
[31] J. W. Jiang, T. Lan, S. Ha, M. Chen, and M. Chiang, "Joint VM placement and routing for data center traffic engineering," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 2876–2880.
[32] T. Chen, X. Gao, and G. Chen, "Optimized virtual machine placement with traffic-aware balancing in data center networks," *Sci. Program.*, vol. 2016, Aug. 2016, Art. no. 3101658, doi: 10.1155/2016/3101658.
[33] F. Yan, T. T. Lee, and W. Hu, "Congestion-aware embedding of heterogeneous bandwidth virtual data centers with hose model abstraction," *IEEE/ACM Trans. Netw.*, vol. 25, no. 2, pp. 806–819, Apr. 2017.
[34] J. Son and R. Buyya, "Priority-aware VM allocation and network bandwidth provisioning in software-defined networking (SDN)-enabled clouds," *IEEE Trans. Sustain. Comput.*, to be published, doi: 10.1109/TSUSC.2018.2842074.
[35] R. Chi, Z. Qian, and S. Lu, "Be a good neighbour: Characterizing performance interference of virtual machines under Xen virtualization environments," in *Proc. IEEE Int. Conf. Parallel Distrib. Syst. (ICPADS)*, Dec. 2014, pp. 257–264.
[36] F. Xu, F. Liu, and H. Jin, "Heterogeneity and interference-aware virtual machine provisioning for predictable performance in the cloud," *IEEE Trans. Comput.*, vol. 65, no. 8, pp. 2470–2483, Aug. 2016.
[37] H. Zhao, J. Wang, F. Liu, Q. Wang, W. Zhang, and Q. Zheng, "Power-aware and performance-guaranteed virtual machine placement in the cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 6, pp. 1385–1400, Jun. 2018.
[38] F. Xu, F. Liu, L. Liu, H. Jin, B. Li, and B. Li, "iAware: Making live migration of virtual machines interference-aware in the cloud," *IEEE Trans. Comput.*, vol. 63, no. 12, pp. 3012–3025, Dec. 2014.
[39] D. Li, J. Wu, and W. Chang, "Efficient cloudlet deployment: Local cooperation and regional proxy," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Mar. 2018, pp. 757–761.



XIN LI (M'17) received the B.S. and Ph.D. degrees from Nanjing University in 2008 and 2014, respectively. He is currently an Assistant Professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. His research interests include computer networking, cloud computing, and data management.

ZHEN LIAN is currently pursuing the master's degree with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. His research interests include cloud computing and data management.

XIAOLIN QIN is currently a Professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. His research interests include data management and knowledge discovery.

JIE WU received the Ph.D. degree in computer engineering from Florida Atlantic University, Boca Raton, FL, USA, in 1989. He is currently the Chair and the Laura H. Carnell Professor with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA, USA. Prior to joining Temple University, he was the Program Director with the National Science Foundation and a Distinguished Professor with Florida Atlantic University. He has regularly published in scholarly journals, conference proceedings, and books. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. He received the 2011 CCF Overseas Outstanding Achievement Award. He was the General Co-Chair/Chair of the 2006 IEEE International Conference on Mobile Ad hoc and Sensor Systems and the 2008 IEEE International Parallel and Distributed Processing Symposium and the Program Co-Chair of the 2011 IEEE International Conference on Computer Communications. He also served as the General Chair for the 2013 IEEE International Conference on Distributed Computing Systems and the 2014 Association for Computing Machinery (ACM) International Symposium on Mobile Ad Hoc Networking and Computing and the Program Chair for the China Computer Federation (CCF) China National Computer Congress 2013. He was an IEEE Computer Society Distinguished Visitor, an ACM Distinguished Speaker, and the Chair of the IEEE Technical Committee on Distributed Processing. He is also the CCF Distinguished Speaker. He serves on several editorial boards, including the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON SERVICES COMPUTING, and the *Journal of Parallel and Distributed Computing*.