J Inf Process Syst, Vol.16, No.2, pp.460~477, April 2020
https://doi.org/10.3745/JIPS.02.0112

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Energy-Aware Virtual Data Center Embedding

Xiao Ma*, Zhongbao Zhang*, and Sen Su*

Abstract

As one of the most significant challenges in the virtual data center, the virtual data center embedding has attracted extensive attention from researchers. The existing research works mainly focus on how to design algorithms to increase operating revenue. However, they ignore the energy consumption issue of the physical data center in virtual data center embedding. In this paper, we focus on studying the energy-aware virtual data center embedding problem. Specifically, we first propose an energy consumption model. It includes the energy consumption models of the virtual machine node and the virtual switch node, aiming to quantitatively measure the energy consumption in virtual data center embedding. Based on such a model, we propose two algorithms regarding virtual data center embedding: one is heuristic, and the other is based on particle swarm optimization. The second algorithm provides a better solution to virtual data center embedding by leveraging the evolution process of particle swarm optimization. Finally, experiment results show that our proposed algorithms can effectively save energy while guaranteeing the embedding success rate.

Keywords

Energy-Aware, Particle Swarm Optimization, Virtual Data Center Embedding, Virtual Link, Virtual Node

1. Introduction

Virtual data center (VDC) is a new form of cloud computing applied to the data center. VDC abstractly integrates physical resources through virtualization technology, thus dynamically allocating and scheduling resources. There are two benefits: sharing resources of physical data center among multiple users, and greatly reducing the operating costs of data centers [1].

There are two roles existing in VDC: infrastructure provider (InP) and service provider (SP). The former controls the infrastructure that runs in the entire data center. The latter creates its VDC above the infrastructure and deploys corresponding services and applications for end users. Each VDC embedding request has a network topology consisting of a certain amount of virtual nodes and virtual links with resource requirements. And there are two types of virtual nodes: virtual machine nodes and virtual switch nodes, which have demands for resources (like CPU, memory and hard disk) and switch ports, respectively. Virtual links often represent the need for communication bandwidth. When InP provides a VDC embedding request service for an SP, it needs to find nodes and links that meet these resource requirements in the physical data center network, and then allocate corresponding resources for the deployment of protocols or services, etc. This is termed as the problem of VDC embedding.

Manuscript received August 9, 2018; first revision November 14, 2018; accepted January 2, 2019; onlinefirst July 29, 2019.

Corresponding Author: Xiao Ma (max106485@bupt.edu.cn)

* State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunication, Beijing, China (max106485@bupt.edu.cn, zhongbaozb@bupt.edu.cn, susen@bupt.edu.cn)

VDC embedding has always been a hot topic in recent researches. Existing research work focuses on how to enable InP to accept more VDC embedding requests, or to provide VDC embedding with fixed bandwidth for offering better service guarantees. However, the energy consumption of current InP during its operation has not received sufficient attention. For example, in China, China Mobile Communications Corporation needs trillions of watt-hours of electricity per year, which is equal to the annual electricity consumption of tens of millions of households. This shows that the issue of energy consumption in the Internet industry cannot be ignored. Therefore, to maximize profits, InP also needs to minimize energy consumption to reduce operating costs.

There are three challenges in achieving the goal of minimizing energy consumption.

The first challenge is how to model and quantify energy consumption. This paper first classifies physical nodes into two categories: server nodes and switch nodes. The former is used to perform computing tasks; while the latter is used to transmit and forward communication data between physical nodes. Then according to the state of the nodes, they need to be further divided into powering-on nodes and powering-off nodes. However, changing from the unopened mode to the opened mode requires a part of additional energy consumption overhead. This paper establishes a corresponding energy consumption model for different types of nodes, then performs different quantitative analyses based on the model, and finally calculates the entire energy consumption of physical nodes during the VDC embedding process.

The second challenge is how to design an energy-aware VDC embedding algorithm. To address this challenge, this paper designs a corresponding heuristic algorithm. The algorithm is divided into two steps: virtual node embedding and virtual link embedding. In the former process, there are two sub-steps: virtual switch node embedding and virtual machine node embedding. In the process of the virtual switch node embedding, the virtual switch node is preferentially embedded to a position close to the server node according to the characteristics of physical data centers. Firstly, for the purpose of improving the success rate of virtual link embedding, we utilize the worst-fit strategy to sort the physical switches that satisfy the link resource demand, on the basis of the ingress and egress link bandwidth resources of the physical switches. Secondly, we add labels for the physical switch nodes, and these labels represent the amount of available underlying virtual machine resources. Thirdly, for the purpose of ensuring the utilization of the node resources, we employ the best-fit strategy to sort the physical switch nodes that meet the virtual machine resource demands with the help of labels we have added. Finally, we add the above two sorted sequence to find the top physical switch node as the target node for the virtual switch node embedding. In the process of the virtual machine node embedding, we directly find the suitable physical server node in the lower layer according to the embedding place of the virtual switch node. In the process of virtual link embedding, we use the shortest path strategy. The core idea is to select the lowest number of powering-off nodes on the path that is preferentially selected. Through the above embedding processes, the goal of energy saving will eventually be achieved.

The third challenge lies in that the heuristic algorithm above merely uses the heuristic information to find the only solution, and its performance is still waiting to be optimized. Thus, we use the population-based optimization technique to find multiple solutions and then select the best one. Through the existing population-based techniques, particle swarm optimization (PSO) is widely applied in lots of fields, since it has strong robustness and faster execution and higher efficiency. We leverage this technique for the VDC embedding problem. In this algorithm, we redefine particle related parameters and operations in Section 5.3 and then propose the best-fit and worst-fit hybrid strategies.

We summarize the contributions as follows:

1) This paper studies energy consumption in the VDC embedding process. To reduce the consumption, we propose the energy consumption model for switch nodes and server nodes, and design an algorithm of energy-aware VDC embedding.
2) This paper further proposes a PSO-based VDC embedding algorithm. In the algorithm, we redefine particle related parameters and operations in Section 5.3 and then propose the best-fit and worst-fit hybrid strategy.
3) This paper evaluates the energy-aware VDC embedding algorithms through lots of simulation experiments. The results show that the two proposed algorithms can greatly reduce the energy consumption in the VDC embedding process while guaranteeing the embedding success rate.

The following sections of this paper are arranged as follows. Section 2 summarizes the related research works. Section 3 gives the description of the VDC embedding problem, and proposes the energy consumption model of switch nodes and server nodes. In Sections 4 and 5, we propose a heuristic and PSO-based algorithm for energy-aware VDC embedding, respectively. In Section 6, the experiment is designed by comparing the analysis results and evaluating the efficiency of the embedding algorithm. Finally, Section 7 gives the summarization of this paper.

2. Related Work

For the past few years, VDC embedding has become more and more important in the field of virtualization technology research and received widespread attention from experts and scholars. It is related to virtual machine placement and virtual network embedding. These issues are described as follows.

Virtual machine placement (VMP): Virtual machine placement refers to the problem of placing virtual machines with node requirements in the data center. The authors [2] mainly solves the problem of resource waste caused by excessive resource fragmentation in the data center. According to the multi-path transmission and virtual machine migration technology, two algorithms are proposed. These two algorithms can respectively deal with the static and dynamic network environments and effectively improve the data utilization rate. Based on the specific conditions of different applications, especially the applications deployed in multi-level virtual machines that require frequent communication, the authors [3] proposes an application-aware virtual machine migration strategy. It ensures that when the virtual machine migrates to relieve the load of the data center, the normal operation of these applications will not be affected. However, unlike VDC embedding, it does not consider the allocation of bandwidth resources on the links.

Virtual network embedding (VNE): There are multiple virtual networks on a physical network. These virtual networks dynamically share physical nodes and physical links on a physical network under the constraints of certain service level agreements (SLAs) and resources. In [4-6], the energy consumption model of nodes and links is proposed, and an energy-aware VNE algorithm is designed to effectively reduce the energy consumption of nodes and links during virtual network embedding. Several studies [7-9] consider the dynamic nature of virtual network embedding and design a VNE algorithm in a dynamic environment.

Although VDC embedding and VNE are somewhat similar, there are three differences between them. The first point lies in the scales. There are usually thousands of nodes in the physical data center while

only 50–100 nodes in the physical network. Secondly, they have different types of topology. The physical data center usually adopts some hierarchy topology (e.g., tree) while the physical network usually adopts some flat topology. Thirdly, there are more than one kind of nodes in the physical data center, e.g., switch node and server node. The three differences above make the problem of VDC embedding so difficult that the existing VNE algorithm is not applicable.

Virtual data center embedding (VDCE): Current research works about VDC networks mainly focus on the aspects of packet forwarding policy, bandwidth control guarantee, multi-path transmission, and bandwidth sharing. Diverter [10] uses the software (VNET) to simplify the work of forwarding packets by switches and routers. However, it does not consider the bandwidth problem in the communication of nodes. VICTOR [11] uses the strategy of migrating virtual machines to complete the load balancing problem in the data center network, but it requires the network equipment to support many specific protocols with large changes. Oktopus [12] defines two types of VDC embedding request and proposes a bandwidth guarantee embedding algorithm. However, its requirements for the network topology of the physical data center are special, and its applications are limited. SecondNet [13] controls the completion of VDC embedding through the VDC Manager, and it provides three levels of embedding schemes to meet different embedding requirements from different users. However, it is not applicable to data center networks of many types of topology. For example, it increases the utilization of network resources in BCube [14] topological type, but reduces the one in VL2 [15] and fat-tree. CloudNaaS [16] adopts a programmable approach to add a logical layer on the data center network. It is similar to the software defined network (SDN) that controls the network nodes together. The disadvantage of this method is the difficult operation and the increase in the new network communication overhead. Moreover, they do not consider energy consumption in the process of scheduling resources in the data center.

Recently, based on the balance between operator revenue, energy consumption and carbon emissions, Greenhead [17] suggests first partitioning the VDC requests and then completing the embedding of multi-domain physical data centers. Different from this paper, its data center spans multiple areas and is interconnected through the backbone network of the NSFNet topology. After the VDC embedding request is divided, and the communication links between them need to be embedded to the backbone. How to properly embed the above links on the backbone network has become an important method to achieve the energy-aware goals. This issue is fundamentally different from this paper, so the two are not compared.

Yang et al. [18] propose a robust green VDC embedding solution. Specifically, a Nearest-edge-Switch approach is used to embed virtual Servers (NSS) and then a Joint embedding approach is to embed virtual Switches and Links (JointSL). This is a heuristic solution; however, our proposed solution can leverage the evolution process of PSO to get a better solution, as demonstrated through extensive experiments in Section 6.2. Gilesh et al. [19] propose a novel solution, which can minimize fragmentation to achieve a higher acceptance rate compared to existing strategies; meanwhile, it minimally disrupts the existing VDCs. However, it still ignores the energy consumption, which hence is excluded from the comparison.

3. Description of VDC Embedding

This section includes problem descriptions, and the introduction of the energy consumption model and evaluation metrics.

3.1 Problem Description

3.1.1 Physical data center

As an infrastructure provided by InP, there are many types of physical data center topology [1]. In the traditional binary tree structure, the root node is easy to make a communication bottleneck. The path between the nodes in the fat-tree structure gradually widens from the leaf node, adapting the traffic from the leaf node to the root node in order to make the use of resources in data center networks more convenient. Therefore, this paper mainly uses the above advantages of fat-tree structure to define the network topology of the physical data center. Its concrete topological structure is shown in Fig. 1(d). It can be represented by a weighted undirected graph $G^p(N^p, S^p, L^p)$, where N^p is the set of physical server nodes, S^p is the set of physical switch nodes, and L^p is the set of links between nodes.

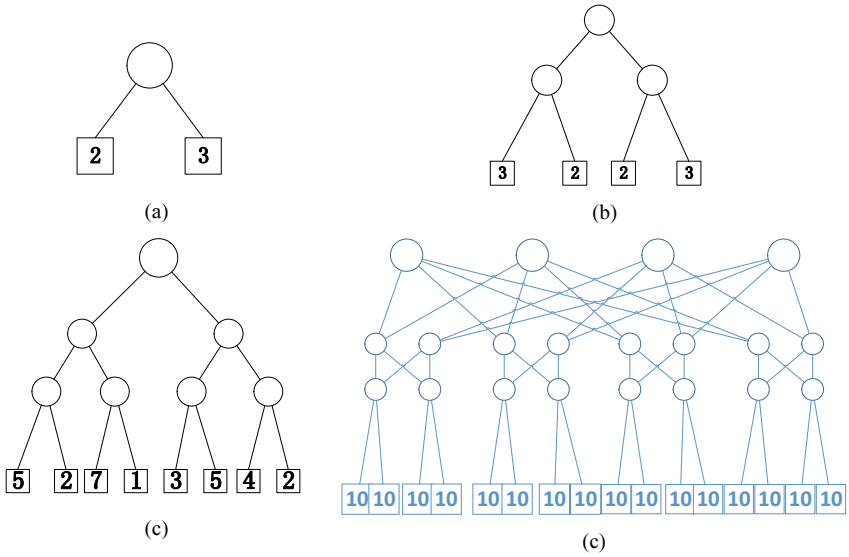

Fig. 1. VDC request + physical data center network topology: (a) one layer VDC request, (b) two layers VDC request, (c) three layers VDC request, and (d) fat-tree physical data center network topology.

At each physical server node, there are specific resources such as CPU, memory, and hard disk. However, in this paper, due to the resource constraint on the node, we mainly consider the number of virtual machines that are embedded, as shown in the square node in Fig. 1. We assume that each physical server node has the same configuration capability. For the physical switch node, the above resources are defined as the number of available ports, and each physical switch node is defined as a unified specification. Each physical link allocates certain bandwidth resources for the communication requirements between the nodes. Owing to different types of VDC embedding requirements, it is required to allocate corresponding resources to them and perform related records while embedding and releasing resources. To achieve this goal, for any server node $n \in N^p$, $m(n)$ indicates the amount of currently available virtual machines. For any switch node $s \in S^p$, the variable $m(s)$ indicates the total number of available virtual machines currently on all server nodes connected to it. For any switch node $s \in S^p$, $p(s)$ indicates the number of port resources available on it. Finally, for any physical link in the link set $l \in L^p$, $b(l)$ denotes the current amount of available bandwidth resources.

3.1.2 Virtual data center

VDC mainly includes two parts: virtual nodes and virtual links. The former is divided into two types: virtual switch nodes and virtual machine nodes. As with the physical data center, a VDC request can also be represented by a weighted undirected graph $G^v(N^v, S^v, L^v)$, where N^v indicates the set of virtual machines, S^v indicates the set of virtual switches, and L^v indicates the set of virtual links. For each virtual machine node $v \in N^v$ and VDC embedding request, $m(v)$ indicates its requirement for the number of virtual machine resources. For any of the virtual switch nodes $u \in S^v$ in the request, $m(u)$ indicates the sum of the virtual machine resources required by all underlying virtual machine nodes. For any of the virtual switch nodes in the request $u \in S^v$, we use $p(u)$ to indicate the number of port resource requirements on the physical switch. Finally, for any virtual link $k \in L^v$ in the request, $b(k)$ indicates its required number of bandwidth resources.

This paper defines its topology as a basic binary tree. According to the switch hierarchy, it can be divided into VDC embedding request with one layer, VDC embedding request with two layers and VDC embedding request with three layers. As shown in Fig. 1(a)–(c), circles represent virtual switch nodes, squares represent virtual machine nodes, numbers in square nodes represent the numbers of virtual machine resources required, and connections between nodes represent virtual links between nodes.

3.1.3 Virtual data center embedding

The formal definition of the VDC embedding process is given below. Based on the above-modeled representations of the physical data center and the VDC, VDC embedding refers to M:$G^v(N^v, S^v, L^v) \rightarrow G^p(N^p, S^p, L^p)$. M contains three separate maps as follows.

Virtual machine node embedding is defined by $M_n: N^v \rightarrow N^p$. It refers to embedding each virtual machine node to a corresponding server node of a physical data center on the premise of satisfying the constraints of the virtual machine node.

Virtual switch node embedding is defined by $M_s: S^v \rightarrow S^p$. It refers to the embedding of each virtual switch node to the corresponding physical switch node of the physical data center on the premise of satisfying the constraints of the virtual switch node.

Link embedding is defined by $M_l: L^v \rightarrow L^p$. It refers to embedding each virtual link to the corresponding physical path in the physical data center on the premise of satisfying the virtual link requirement for each physical link on this path.

3.2 Energy Consumption Model

The energy consumption model mainly includes the energy consumption of the physical switch node and the physical server node in the physical data center. Before calculating the energy consumption of these two parts, it is necessary to express the energy consumption of physical switch nodes and physical server nodes.

Energy consumption of the physical switch node: After the virtual switch node $u \in S^v$ is embedded to the physical switch node $s \in S^p$, the extra energy consumption of the physical switch node s needs to be calculated, represented by ΔPS_s^u. The energy consumption of a typical switch is usually related to system throughput, communication loads, its inherent fans and other refrigeration equipment. One of these is the inherent energy consumption overhead of the switch, which is denoted by P_f. A switch uses

multiple ports to connect different devices. These ports are responsible for the communication between the ingress and egress packets of switches. Energy consumption of per port is represented by P_{port}. Based on the above analysis, the total energy consumption of a physical switch is:

$$PS_s = P_f + P \cdot P_{port}, \tag{1}$$

where P indicates the number of currently used ports of the physical switch node s.

Therefore, in order to embed the virtual switch node u, the extra energy consumption of the physical switch node s can be calculated by the following formula:

$$\Delta PS_s^u = \begin{cases} P_f + \Delta P \cdot P_{port} (open\ a\ new\ node) \\ \Delta P \cdot P_{port} \end{cases}. \tag{2}$$

Energy consumption of the physical server node: After embedding the virtual machine node $v \in N^v$ to the physical server node $n \in N^p$, the extra energy consumption of the physical server node n needs to be calculated, which is denoted by ΔPN_n^v. Similar to the energy consumption of the switch nodes, the server nodes also have a fixed part of the basic energy consumption to maintain the normal operation of the machine. In addition, the energy consumption of other parts is mainly linearly related to the CPU loads. Therefore, this paper calculates the energy consumption of physical server node n by the following formula:

$$PN_n = P_b + P_l \cdot \alpha, \tag{3}$$

where P_b represents the energy of the server node with no load, called the basic energy consumption; α represents the current CPU loads, and P_l represents the linear parameter of the server node with α.

Based on the above energy consumption model, to embed the virtual machine node v, the extra energy required by the physical server node n is:

$$\Delta PN_n^v = \begin{cases} P_b + P_l \cdot C_v (open\ a\ new\ node) \\ P_l \cdot C_v \end{cases}, \tag{4}$$

where C_v represents the CPU demand of the virtual machine node v.

Based on the energy consumption model of the physical switch node and the physical server node, the entire energy consumption of the physical data center during the virtual node embedding process is:

$$\Delta E_{node} = \sum_{u \in S^v} \sum_{s \in S^p} x_s^u \cdot \Delta PS_s^u \int_{t_a}^{t_e} dt + \sum_{v \in N^v} \sum_{n \in N^p} y_n^v \cdot \Delta PN_n^v \int_{t_a}^{t_e} dt, \tag{5}$$

where x_s^u and y_n^v respectively represent whether a certain virtual node is embedded on the corresponding physical node. If so, the value is 1; otherwise, the value is 0. And t_a and t_e represent the arrival time and departure time of the VDC embedding request, respectively.

3.3 Evaluation Metrics

This paper mainly considers two evaluation metrics, including the embedding success rate and energy consumption.

The embedding success rate refers to the rate of successfully embedding requests to all embedding requests. It can be defined as follows:

$$\frac{N_s}{N_{all}}, \tag{6}$$

where N_s represents the number of VDC requests that are accepted, and N_{all} represents the amount of VDC requests that need to be embedded.

Then the long-term average energy consumption of the physical data center network is defined as follow:

$$\lim_{T\to\infty} \frac{\sum_{i=1}^{N_s} \Delta E_{node}^{i}(G^v)}{T}, \tag{7}$$

where N_s represents the number of VDC requests that are successfully embedded during the time T, and $\Delta E_{node}^{i}(G^v)$ represents the energy consumption caused by embedding the i^{th} VDC embedding request.

4. Energy-Aware VDC Embedding Algorithm

For the purpose of solving the energy consumption problem in the VDC embedding process, this paper proposes an energy-aware embedding algorithm, which is divided into following three steps:

1) Node label algorithm based on available resources. We add two variable parameters for each switch node in the physical data center network, which respectively represent the amount of available virtual machines of the underlying network topology and the bandwidth resources available for connecting the switch nodes.
2) Energy-aware virtual node embedding algorithm. According to the network topology of the VDC embedding request, we analyze the root node that needs to be embedded and the corresponding levels of the physical data center network. With the help of the labels we have already added, we then use the best-fit strategy in node resource selection and the worst-fit strategy in bandwidth selection. According to the above two strategies, we find the most suitable embedding target for the root node, and then design a recursive algorithm to complete the embedding of the remaining nodes.
3) Energy-aware virtual link embedding algorithm. The virtual link embedding is performed by using the energy-saving shortest path method to save energy cost while ensuring the link requirements of the VDC.

4.1 Node Label Method Algorithm on Available Resources

Since the physical data center network topology is defined as fat-tree in this paper, each layer contains many switch nodes (except for leaf nodes). This paper uses the node label method based on available resources to add two attribute variables to each switch node as follows: one represents the number of available virtual machine node resources on all network nodes that are connected to the lower part of the switch node (as shown in Fig. 2(d)). The other represents available bandwidth resources outward from the node. When a VDC embedding request comes, based on the above two attribute variables, those nodes

that do not meet the embedding request can be quickly eliminated, thereby providing support for the subsequent specific embedding algorithm.

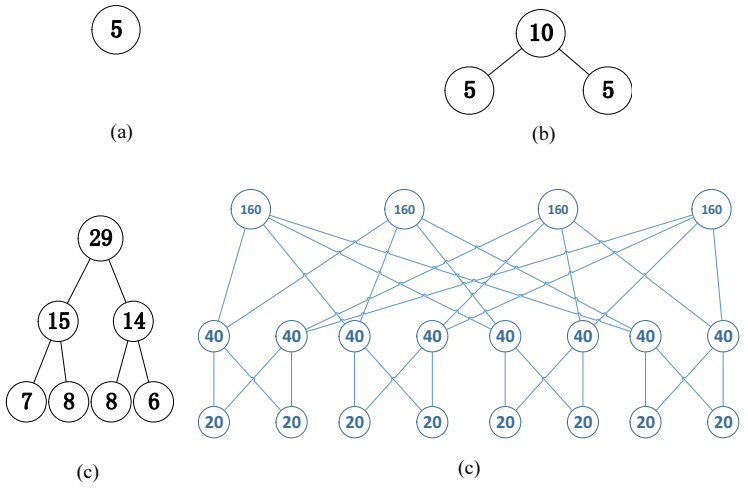

Fig. 2. VDC request with labels + physical data center with labels: (a) one layer VDC request with labels, (b) two layers VDC request with labels, (c) three layers VDC request with labels, and (d) physical data center network topology with labels.

4.2 Energy-Aware Virtual Node Embedding Algorithm

According to the node labeling method based on the available resources in the previous step, the corresponding node embedding algorithm needs to be designed for energy saving. Considering the topology structure of the VDC embedding request, the corresponding network hierarchy is sought to ensure that the nodes in the embedding request correspond to the nodes in the physical data center network. Since the resource requirements of each node and each link in the VDC request have already been defined when it arrives, referring to the previous step, the request-based node label method is used. For each virtual switch node in the VDC request, we add two attribute variables to represent the number of virtual machine node resources required by its lower layer (as shown in Fig. 2(a)–(c)) and the number of bandwidth resource requirements, respectively.

We design a recursive embedding algorithm to complete the embedding task. Inspired by the packing algorithm and the number of node resource requirements at the lower level of the root node in the request, we use the best-fit strategy to sort all switch nodes at the corresponding level of the physical data center network, find the most suitable embedding position and improve the resource utilization rate.

In order to explain the best-fit strategy better, the following section describes the packing problems. Classic packing problems require that a certain amount of items are placed in boxes with the same capacity, so that the sum of the items in each box will not exceed the capacity of the box. The bin packing problem is a complex discrete combinatorial optimization problem. The so-called combinatorial optimization means finding a solution that satisfies a given condition and makes its target function value maximum or minimum on a discrete, finite mathematical structure.

Depending on different resource requirements, the virtual nodes are compared to items of different sizes. Then the physical nodes are compared to the boxes in use, according to the current resource usage.

Based on the resource requirements of the virtual nodes and the available resources of all physical nodes, we calculate the number of remaining available resources to decide whether the virtual nodes can be embedded on the physical nodes and then sort them. The best-fit strategy is to embed the virtual nodes on the physical node that can fulfill them with the fewest remaining resources, thus ensuring the optimal utilization of node resources.

According to the ingress and egress bandwidth requirements of virtual nodes and the available ingress and egress bandwidth resource on all physical nodes, we calculate the number of remaining available bandwidth resources of the physical nodes after embedding the virtual nodes, and then sort candidate physical nodes. The worst-fit strategy is used to find the most suitable embedding position. The worst-fit strategy is to embed the virtual nodes on the physical node that can fulfill them with the most available ingress and egress bandwidth resources. Compared with the best-fit strategy, its resource utilization rate is lower, and the success rate of subsequent embedding is higher.

Finally, we add the above two sorted sequences together. According to the result after addition, the top-ranked node is taken as the most suitable position where the virtual node is to be embedded. This approach not only achieves the goal of energy saving, but also facilitates other follow-up embedding requests. For other nodes in the embedding request, the embedding is done in the same way. The specific algorithm is shown in Algorithm 1.

Algorithm 1: Energy-aware virtual node embedding algorithm

Input: VDC embedding request $\boldsymbol{G_v}$, physical data center network $\boldsymbol{G_p}$.
Output: Virtual Node embedding Scheme.

```
1.  embedding ( node in G_v )
2.    If ( node==NULL )
3.      Return successful embedding ;
4.  Define the hierarchy of variable d=node in node G_v.
5.  Define the total number of layers of network topology in variable n=G_p.
6.  For ( all nodes in the n-d+1 layer of G_p )
7.    Obtain rank NR based on best-fit strategy on        available node resources.
8.    Obtain rank LR according to worst-fit strategy on the number of available bandwidth resources.
9.    Define the variable R=NR+LR (final rank).
10.   Sort R from small to large.
11.     If ( all items in R are infinite )
12.       Return node embedding failed
13.     Else
14.       If ( node has a child node )
15.         For ( every child belongs to the node )
16.           embedding ( child of node )
17.   Map the node to the first position in R.
```

4.3 Energy-Aware Virtual Link Embedding Algorithm

Virtual link embedding means that, a non-recursive path should be found between physical nodes in the corresponding physical data center network after virtual node embedding, and all the links on the path

should satisfy the bandwidth resource requirements of the corresponding virtual link. In this process, existing studies usually use the shortest path method to solve the problem, but neglect the on/off state of physical data center network nodes, leading to increased energy consumption with too many opened nodes. Thus, this paper designs an energy-aware virtual link embedding algorithm. First, in the topology diagram of the physical data center network, the physical links that do not satisfy the bandwidth requirements of the virtual link to be embedded are deleted, and the remaining graphs are called residual figures. Then, in the residual figures, we calculate the shortest path. It is possible to get multiple shortest paths here. Finally, we choose paths that have the least number of opened nodes among the multiple shortest paths. By using the algorithm, we not only select the shortest path to avoid the long path problem of link embedding, but also reduce the number of nodes that are newly opened to achieve the goal of energy-aware. The specific algorithm is is shown in Algorithm 2.

Algorithm 2: Energy-aware virtual link embedding algorithm

Input: VDC embedding request $\boldsymbol{G_v}$, physical data center network $\boldsymbol{G_p}$ and virtual node embedding scheme.
Output: Virtual Link embedding Scheme.

1. For every virtual link $\boldsymbol{l_v}$ belongs to $\boldsymbol{G_v}$
2. Delete all physical links in the physical data center network $\boldsymbol{G_p}$ that cannot meet the resource requirements of the virtual link $\boldsymbol{l_v}$.
3. In the residual network, find the corresponding shortest path set.
4. In this set, find the path with the smallest number of newly opened nodes P.
5. If path P can meet all $\boldsymbol{l_v}$ bandwidth resource requirements then
6. Map $\boldsymbol{l_v}$ to path P and allocate bandwidth resources for $\boldsymbol{l_v}$.
7. Else
8. Release bandwidth resources of allocated physical paths.
9. Return virtual link embedding failed
10. Return virtual link embedding succeeded

5. Energy Aware Particle Swarm Optimization based Virtual Data Center Embedding (EA-VDCE-PSO)

In the last section, we propose a heuristic VDC embedding algorithm. However, its performance is still waiting to be optimized. It only uses heuristic information to find only one solution. Next, we use the population-based optimization technique to find multiple solutions and then select the best solution. Through the existing population-based techniques, PSO is widely applied in lots of fields, because of its strong robustness, faster execution, and higher efficiency. Thus, we leverage this technique to VDC embedding problem. In this section, we introduce PSO basis in Section 5.1, present two challenges to adopt PSO in our problem in Section 5.2, propose two strategies for these two challenges in Sections 5.3 and 5.4, and finally give the description of this algorithm in Section 5.5.

5.1 PSO Basis

The PSO algorithm is based on swarm intelligence [20]. In this algorithm, each particle moves in the solution space at some speed, and aggregates to its personal historical best position X_{pb} and the global

historical optimal position X_{gb} to realize the evolution of the candidate solution. The update formula of the particle's velocity and position is as follows:

$$V_{i+1} = \omega V_i + c_1 r_1 (X_{pb} - X_i) + c_2 r_2 (X_{gb} - X_i), \tag{8}$$

$$X_{i+1} = X_i + V_{i+1}, \tag{9}$$

where X_i represents the current position of the i-th particle and V_i represents its current velocity; w represents the weight of the particles holding inertia; c_1 and c_2 as the acceleration of the particles represent the tendency of the particles to move to their historical best position and global best position respectively; r_1 and r_2 are uniformly generated random numbers between (0, 1).

5.2 Challenges

When applying PSO into the energy-aware VDC embedding problem, there are two challenges as below:

1) VDC embedding is a discrete problem. Thus, how to make PSO suitable for discrete optimization problem?
2) In basic PSO, it randomly chooses the position in the solution space, which may lead to the slow convergence in VDC embedding.

To address these two challenges, we first redefine particle-related parameters and operations in Section 5.3, and then propose the best-fit and worst-fit hybrid strategy in Section 5.4.

5.3 Redefinition of Particle Related Parameters and Operations

PSO is mainly used to settle some problems in continuous domains. While solving discrete optimization problems, it is necessary to redefine the parameters and related operations of particles on the basis of specific problems. According to the optimized model of VDC embedding, we redefine the position, velocity and related operations in the particle swarm, as below:

DEFINITION 1. The position of one particle: The position vector $X_i = [x_i^1, x_i^2, \cdots, x_i^D]$ is defined as the i^{th} possible mapping scheme. D indicates that the virtual network request contains a total of D VDC nodes; x_i^j takes a positive integer whose value represents the number of nodes in the physical data center selected by the j^{th} VDC node from its list of underlying network candidate nodes.

DEFINITION 2. The velocity of two particles: The velocity vector $V_i = [v_i^1, v_i^2, \cdots, v_i^D]$ of the particle is defined by the adjustment decision of the mapping scheme, which guides the current mapping scheme to the better mapping scheme. Here, v_i^j is a binary variable. If $v_i^j = 0$, it means that the j^{th} virtual node needs to reselect the node mapping from its candidate node list of the physical data center.

DEFINITION 3. Subtraction Θ: $X_i \Theta X_j$ is used to calculate the difference between the two mapping schemes. If the mapping schemes X_i and X_j have the same value in the same dimension, the result of the difference is 1, otherwise 0. For example, $(1,2,3,4,5)\ \Theta(1,8,7,4,9) = (1,0,0,1,0)$.

DEFINITION 4. Addition ⊕: $P_iV_i \oplus P_jV_j$ is used to obtain the adjustment decision of the mapping scheme. P_iV_i and P_jV_j respectively represent that the values of V_i dimensions are maintained with the probability of P_i and the values of V_j are maintained with the probability of P_j, and $P_i + P_j = 1$ $(0 \leq P \leq 1)$. For example, 0.3(1,0,0,1,1) ⊕ 0.7(1,0,1,0,1) = (1,0,*,*,1), where * denotes that this dimension 0 or 1 is uncertain. In addition, the first * indicates that this dimension takes 0 with a possibility of 0.3 and takes 1 with a possibility of 0.7.

DEFINITION 5. Multiplication ⊗: $X_i \otimes V_i$ is used to obtain a new mapping scheme. The mapping scheme X_i adjusts its virtual node mapping scheme according to the adjustment decision V_i. For example, $(1, 8, 5, 9, 10) \otimes (1, 0, 1, 1, 1)$, indicating that the mapping scheme of the second virtual node should be adjusted.

Therefore, we can derive the basic formula for the position and velocity update of the redefined PSO algorithm as follows:

$$V_{i+1} = P_1V_i \oplus P_2\left(X_{pb} \Theta X_i\right) \oplus c_2r_2\left(X_{gb} \ominus X_i\right) \tag{10}$$

$$X_{i+1} = X_i \otimes V_{i+1} \tag{11}$$

where P_1, P_2, and P_3 are constants, and $P_1 + P_2 + P_3 = 1$.

5.4 Best-Fit and Worst-Fit Hybrid Strategy

In basic PSO, it randomly chooses the position in the solution space, which may lead to the slow convergence in our problem. To settle this problem, the best-fit and worst-fit hybrid strategy is proposed, which shares the same idea with the heuristic algorithm in Section 4.2. We adopt the best-fit strategy to sort all switch nodes in the physical data center network, find the most suitable embedding position and improve resource utilization rate. Then we use the worst-fit strategy to embed the virtual nodes on the physical node that can fulfill them with the most available ingress and egress bandwidth resources. Finally, we add the above two sorted sequences together and give the final rank. If a physical node has a higher rank, it is more likely to be selected as the candidate embedding node.

This strategy guarantees that the selected node can satisfy the node and link resource constraint, and thus has a faster convergence speed.

5.5 VDCE-PSO Algorithm Description

The VDCE-PSO algorithm makes the number of active physical nodes as the fitness function $f(X)$, where the position vector X represents a possible mapping scheme. If the mapping scheme is feasible, then the value of $f(X)$ represents the overhead of the VDC embedding. If the embedding scheme is not feasible, then the value of $f(X)$ is set to +∞; the description of the VDCE-PSO algorithm is as follows:

Step 1: Set the number of particle swarms to N, and the maximum number of iterations executed by the algorithm to MG. The particle randomly generates the initial position parameter X_i and the velocity parameter V_i.

Step 2: Calculate the fitness $f(X_i)$ of all particles, and obtain the global optimal initial position X_{gb} and the optimal initial position X_{pb} of each particle.

Step 3: Update the velocity of the particles satisfying the constraint according to Eq. (10); update the position in Eq. (11), randomly select the candidate nodes of the physical data center in the position updating process, and regenerate the position and velocity parameters for the particles that do not satisfy the node constraints.

Step 4: For each particle in the particle group, if $f(X_i) < f(X_{pb})$, then $X_{pb} = X_i$; if $f(X_{pb}) < f(X_{gb})$, then $X_{gb} = X_{pb}$.

Step 5: Check the current number of iterations. If it is less than MG, execute Step 3; otherwise, execute Step 6.

Step 6: Output the optimal VDC embedding scheme and its fitness value.

6. Experimental Evaluation

In this section, we assess our proposed two energy-aware VDC embedding algorithms. Firstly, the experimental environment is described. Then the experimental results are analyzed. Compared with other existing works, this paper proves that the energy-aware VDC embedding algorithms can make a significant increment in terms of the embedding success rate and energy consumption.

6.1 Experimental Configuration

Physical data center settings: This paper uses a three-layer fat-tree structure to define the network topology of the physical data center. Each switch has 16 ports. Therefore, in the experimental environment, there are 320 physical switches, 1024 physical servers, and 3072 physical links. Each physical server has 10 units of virtual machine resources at most; while each physical link has 100 units of bandwidth resources.

Virtual data center request settings: For each VDC embedding request, this paper uses a binary tree to define their network topology. As shown in Fig. 1(a)–(c), there are three types: one-layer architecture, two-layer architecture, and three-layer architecture. Each virtual machine has different requirements for the virtual machine resources, and each virtual link needs to occupy one port of the physical switch at both ends of the link. Meanwhile, the resource demands of virtual links for bandwidth resources are between 1 and 10 units.

This paper defines that an average of one VDC embedding request will arrive for per unit of time, and the request type is a random one of the three types, and there are 2000 VDC embedding requests totally. The VDC embedding request is normally distributed during the period of time. At the same time, according to the energy-aware VDC embedding algorithm proposed in this paper, resources should be allocated for each VDC embedding request when it comes while resources should be released when it leaves so as to serve other coming VDC embedding requests.

Compared algorithm: To evaluate our proposed solution, we compare our algorithms to the latest algorithm, NSS-JointSL algorithm, proposed in [18]. In this algorithm, first a Nearest-edge-Switch

approach is used to embed virtual Servers (NSS) and then a Joint embedding approach is employed to embed virtual Switches and Links (JointSL). This is a heuristic solution. However, our proposed solution leverages the evolution process of PSO to get a better solution, which can be demonstrated through an extensive experiment in Section 6.2. Gilesh et al. [19] propose a novel solution. It minimizes fragmentation to achieve more accepted VDC requests compared to existing strategies, meanwhile minimally disrupting the existing VDCs. However, they also ignore the energy consumption. Thus, we exclude the solution from the comparison.

Other research works about VDC embedding problem mainly focus on enhancing operating revenue, improving the embedding success rate and providing bandwidth guarantee. Therefore, to facilitate a fair comparison, we exclude these existing studies and choose the random embedding algorithm (R-VDCE) as the compared algorithm. That is, when a VDC embedding request comes, the physical nodes and physical paths that satisfy the resource needs of both virtual nodes and virtual links are randomly found on the physical data center.

Evaluation metrics: The evaluation metrics include the embedding success rate of VDC embedding requests and energy consumption during the embedding process, as shown in Eqs. (6) and (7).

6.2 Experimental Results and Analysis

Compared to the state-of-the-art algorithm: To evaluate the performance of the EA-VDCE algorithm proposed in this paper, we compare it to the random embedding algorithm R-VDCE and the algorithm NSS-SL [18]. Figs. 3 and 4 show the experimental results. EA-VDCE has a higher successful rate of embedding and less energy consumption than R-VDCE. The reasons are as follows. First, the EA-VDCE algorithm comprehensively considers the scheduling of node resources and bandwidth resources, so it provides a better embedding environment for subsequent VDC embedding requests and improves the embedding success rate. Second, it figures out the type of VDC request (one layer, two layers or three layers) and then maps it to the corresponding layer of the physical data center, which avoids the long-path problem and effectively reduces the energy consumption. Third, the algorithm proposed in this paper adds labels to each physical node. By these labels, we can find more suitable physical nodes during the embedding process as soon as possible. In this way, the number of newly opened physical nodes is decreased and the energy consumption is effectively reduced.

For the success rate, from Fig. 3, through the experiments above, compared with R-VDCE and NSS-SL, EA-VDCE maintains a relatively high success rate of embedding, reaching almost 80%. However, the success rate of R-VDCE is affected by the increase of embedding requests, and the average rate is maintained at about 60%.

For the energy cost, from Fig. 4, we can also see that EA-VDCE, on the average, has reduced energy consumption by about 28% and 11% compared with R-VDCE and NSS-SL. This is because EA-VDCE can consolidate the VDC nodes into a smaller number of physical data center nodes and thus power off more nodes into a sleeping state.

The impact of the PSO-based algorithm: From Fig. 3, for the success rate, EA-VDCE-PSO maintains a relatively high success rate of embedding which is almost 82%, similar to the other three algorithms. From Fig. 4, for the energy cost, we can also observe that the proposed algorithm EA-VDCE-PSO further reduces energy consumption by 9%, compared to our proposed heuristic algorithm EA-VDCE. The reason is that EA-VDCE-PSO can find more solutions by the particles through the iterative

process. It is smart to find more and more optimal solutions through learning from each other particles. Notably, it also consumes some running time, which is omitted due to the page limit. However, the physical data center owner can adjust the iteration times and the number of particles to balance energy efficiency and running time.

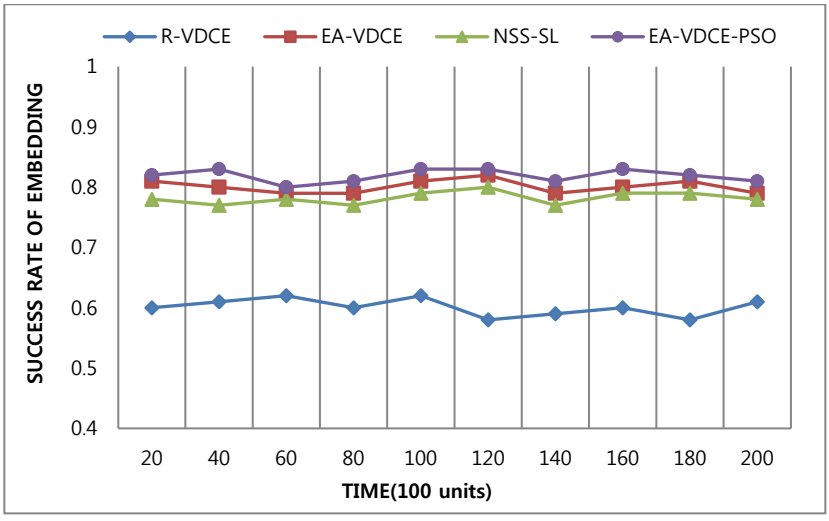

Fig. 3. Comparison of the embedding success rate.

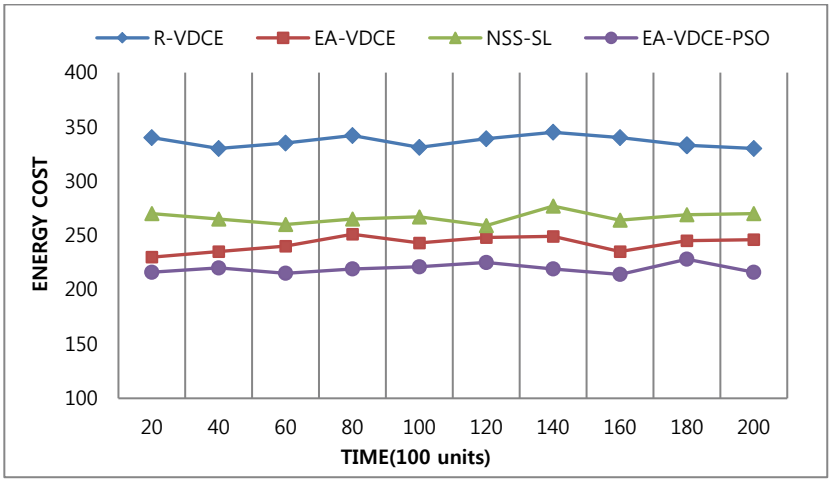

Fig. 4. Comparison of energy cost.

7. Conclusion

To optimize the energy consumption in the embedding process of VDC requests, we propose the energy consumption model of physical nodes, and design two VDC embedding algorithms for energy-saving. The first one is a heuristic algorithm. It computes the label variables of each node, comprehensively uses the best matching strategy, the worst matching strategy and the shortest path algorithm, and then

completes the virtual node and virtual link embedding in turn. The second one is the PSO-based technique. It leverages the smart iterative process to achieve more and more optimal solutions. Simulation experiments show that our proposed two algorithms can greatly reduce the energy consumption with a guaranteed success rate of the VDC request embedding.

Acknowledgement

This work was supported in part by the following funding agencies of China, National Natural Science Foundation (Grant No. 61602050 and U1534201).

References

[1] M. F. Bari, R. Boutaba, R. Esteves, L. Z. Granville, M. Podlesny, M. G. Rabbani, Q. Zhang, and M. F. Zhani, "Data center network virtualization: a survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 2, pp. 909-928, 2012.

[2] J. W. Jiang, T. Lan, S. Ha, M. Chen, and M. Chiang, "Joint VM placement and routing for data center traffic engineering," in *Proceedings of 2012 Proceedings IEEE INFOCOM*, Orlando, FL, 2012, pp. 2876-2880.

[3] V. Shrivastava, P. Zerfos, K. W. Lee, H. Jamjoom, Y. H. Liu, and S. Banerjee, "Application-aware virtual machine migration in data centers," in *Proceedings of 2011 IEEE INFOCOM*, Shanghai, China, 2011, pp. 66-70.

[4] Z. Zhang, S. Su, K. Shuang, W. Li, and M. A. Zia, "Energy aware virtual network migration," in *Proceedings of 2016 IEEE Global Communications Conference (GLOBECOM)*, Washington, DC, 2016, pp. 1-6.

[5] Z. Zhang, S. Su, X. Niu, J. Ma, X. Cheng, and K. Shuang, "Minimizing electricity cost in geographical virtual network embedding," in *Proceedings of 2012 IEEE Global Communications Conference (GLOBECOM)*, Anaheim, CA, 2012, pp. 2609-2614.

[6] S. Su, Z. Zhang, A. X. Liu, X. Cheng, Y. Wang, and X. Zhao, "Energy-aware virtual network embedding," *IEEE/ACM Transactions on Networking*, vol. 22, no. 5, pp. 1607-1620, 2014.

[7] Z. Zhang, S. Su, J. Zhang, K. Shuang, and P. Xu, "Energy aware virtual network embedding with dynamic demands: online and offline," *Computer Networks*, vol. 93, pp. 448-459, 2015.

[8] Z. Zhang, S. Su, J. Zhang, K. Shuang, and P. Xu, "Energy aware virtual network embedding with dynamic demands," in *Proceedings of 2015 IEEE International Conference on Communications (ICC)*, 2015, London, UK.

[9] N. M. K. Chowdhury and R. Boutaba, "A survey of network virtualization," *Computer Networks*, vol. 54, no. 5, pp. 862-876, 2010.

[10] A. Edwards, A. Fischer, and A. Lain, "Diverter: a new approach to networking within virtualized infrastructures," in *Proceedings of the 1st ACM Workshop on Research on Enterprise Networking*, Barcelona, Spain, 2009, pp. 103-110.

[11] F. Hao, T. V. Lakshman, S. Mukherjee, and H. Song, "Enhancing dynamic cloud-based services using network virtualization," in *Proceedings of the 1st ACM Workshop on Virtualized Infrastructure Systems and Architectures*, Barcelona, Spain, 2009, pp. 37-44.

[12] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards predictable datacenter networks," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 242-253, 2011.

[13] C. Guo, G. Lu, H. J. Wang, S. Yang, C. Kong, P. Sun, W. Wu, and Y. Zhang, "Secondnet: a data center network virtualization architecture with bandwidth guarantees," in *Proceedings of the 6th International Conference*, Philadelphia, PA, 2010.

[14] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu, "BCube: a high performance, server-centric network architecture for modular data centers," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, pp. 63-74, 2009.

[15] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, "VL2: a scalable and flexible data center network," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, pp. 51-62, 2009.

[16] T. Benson, A. Akella, A. Shaikh, and S. Sahu, "CloudNaaS: a cloud networking platform for enterprise applications," in *Proceedings of the 2nd ACM Symposium on Cloud Computing*, Cascais, Portugal, 2011.

[17] A. Amokrane, M. F. Zhani, R. Langar, R. Boutaba, and G. Pujolle, "Greenhead: virtual data center embedding across distributed infrastructures," *IEEE Transactions on Cloud Computing*, vol. 1, no. 1, pp. 36-49, 2013.

[18] Y. Yang, X. Chang, J. Liu, and L. Li, "Towards robust green virtual cloud data center provisioning," *IEEE Transactions on Cloud Computing*, vol. 5, no. 2, pp. 168-181, 2015.

[19] M. P. Gilesh, S. M. Kumar, L. Jacob, and U. Bellur, "Towards a complete virtual data center embedding algorithm using hybrid strategy," in *Proceedings of 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, Atlanta, GA, 2017, pp. 2616-2617.

[20] R. Eberhart and J. Kennedy, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, Perth, Australia, 1995, pp. 1942-1948.

Xiao Ma **https://orcid.org/0000-0001-7792-9457**

He is a PhD candidate in the State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications. His current research interests include resource management in data center.

Zhongbao Zhang **https://orcid.org/0000-0002-3242-150X**

He received the Ph.D. degree in Computer Science from Beijing University of Posts and Telecommunications, China, in 2014. He is an assistant professor in Beijing University of Posts and Telecommunications now. His major is Computer Science. His research interests include network virtualization, social network and big data.

Sen Su **https://orcid.org/0000-0003-4266-7527**

He received the Ph.D. degree in Computer Science from the University of Electronic Science and Technology, China, in 1998. He is currently a professor at the Beijing University of Posts and Telecommunications. His research interests include cloud computing and big data.