IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 5, 2020, accepted November 13, 2020, date of publication November 24, 2020,
date of current version December 9, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3040335

VNE-HPSO: Virtual Network Embedding Algorithm Based on Hybrid Particle Swarm Optimization

PEIYING ZHANG[1], YANRONG HONG[1], XUE PANG[1], AND CHUNXIAO JIANG[2,3], (Senior Member, IEEE)

[1]College of Computer Science and Technology, China University of Petroleum (East China), Qingdao 266580, China
[2]Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing 100084, China
[3]Tsinghua Space Center, Tsinghua University, Beijing 100084, China

Corresponding authors: Chunxiao Jiang (jchx@tsinghua.edu.cn) and Peiying Zhang (zhangpeiying@upc.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2020YFB1804800, in part by the National Natural Science Foundation of China under Grant 61922050, and in part by the Major Scientific and Technological Projects of CNPC under Grant ZD2019-183-006.

ABSTRACT Today, the 5G Internet era has arrived, and network virtualization technology has captured the attention of people. However, traditional single-domain virtual embedding technology generated huge overhead in embedding and caused a certain waste of resources. Therefore, for cutting virtual embedding overhead and improving the embedding efficiency, this paper proposes a new multi-domain embedding programme. For this scheme, firstly, to rationalize the use of physical resources to a higher degree, we set up a model for the embedding problem. Secondly, in order to obtain the optimal global solution, we introduce simulated annealing into the PSO algorithm, and distribute the particles through the particle initialization distribution strategy. In addition, for the parameter of inertia weight in the algorithm, we use the linear differential decline strategy to calculate. The experimental results show that compared with VNE-PSO algorithm, this scheme reduces the cost of virtual network mapping by 32.82%, increases the acceptance rate of virtual requests by 7%, and reduces the running time by 18.7%. This scheme maximizes the benefits.

INDEX TERMS Hybrid particle swarm optimization, multi-domain virtual network embedding, embedding cost.

I. INTRODUCTION

Today, the Internet of Things (IoT) technology is increasingly powerful, and network virtualization technology is also advancing with the times. The subject of this research is about virtual network embedding, which is also a pivotal area of network virtualization. Virtual network embedding technology is to distribute the underlying network resources to the virtual network, which is mainly divided into node mapping and link mapping. However, how to allocate is the focus of our research. At present, software-defined network (SDN) is an ideal new network architecture, and it also has a good development prospect in 5G [1]. In the traditional network, forwarding and control functions are implemented by switches or routers at the bottom, while SDN realizes the separation of the forwarding plane and the control plane. This network architecture makes it possible to clearly recognize the underlying network resources and greatly improve the embedding efficiency of virtual network requests. Now, scholars have done a lot of research on the problem of multi-domain embedding. It is mainly because traditional single-domain embedding can no longer meet people's needs. But multi-domain embedding is more practical in the real environment. There are two main difficulties in multi-domain virtual network embedding (VNE) problem: Firstly, due to the structure of the multi-domain network is more complex, the scheme suitable for the single-domain environment is not necessarily suitable for the multi-domain environment. Secondly, due to many reasons such as confidentiality, the local controller does not provide complete domain information, so the VNE done by the global controller is vulnerable. With regard to this problem, we propose a new VNE scheme for the multi-domain problem.

In previous papers on VNE problem, it is very prevailing to use particle swarm algorithm to solve this problem. However, the traditional PSO algorithm has some issues. The solution

The associate editor coordinating the review of this manuscript and approving it for publication was Angelos Antonopoulos.

obtained by this algorithm is not necessarily the optimal global solution. In response to this problem, this paper introduces a new embedding algorithm. The specific innovative work is as follows:

(1) Aiming at the problem of premature convergence of PSO, it is combined with the simulated annealing algorithm. The ability to use simulated annealing to jump at a given probability when finding the optimal position can obtain the optimal solution to a higher degree. In order to understand the two algorithms more vividly, we compared the flow of the PSO algorithm and the SA algorithm, as shown in FIGURE 1.

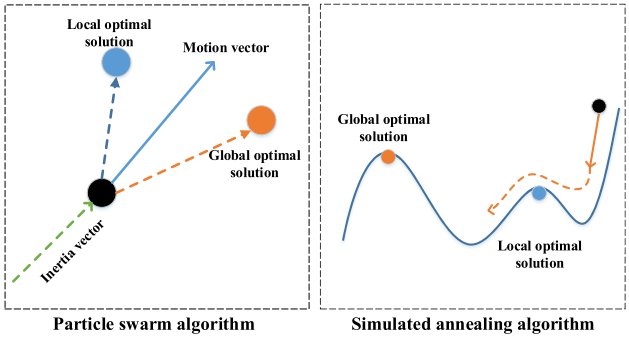


FIGURE 1. PSO and SA optimization process comparison.

(2) For the sake of avoiding limitations of typical linear decline strategy, we use the linear differential decline strategy to calculate the inertia weight. This method accelerates the convergence speed of the embedding algorithm and reduces the embedding overhead to a certain extent.

(3) Aiming at the insufficiency of particle swarm optimization to initialize particle positions, we introduce a particle initialization allocation strategy. Experimental results prove that it can cut down the probability of producing network fragmentation and improve the acceptance rate of virtual requests.

Other works of article arrangements are as follows. Section II retrospects the existing algorithms for solving VNE problem. In Section III, we introduce the details and conduct a model of the VNE problem. In Section IV, we describe the embedding process of the proposed VNE-HPSO algorithm. In Section V, an analysis of experimental results is given. And we summarize all the contents in Section VI.

II. RELATED WORKS

Regarding network virtualization technology, most of the previous papers mainly studied single-domain VNE problem. However, with the expansion of the network scale, researchers have gradually realized the problems of single-domain embedding [2]–[4]. Many existing studies are about multi-domain virtual network embedding [5]. For example, literature [6] points out that virtual requests based on topology can reduce the utilization of physical network to a certain extent. Therefore, this paper proposes a way of traffic matrix to embed virtual requests. Due to the computational complexity of embedding, researchers have tried to find the most practical embedding solution by studying a large number of heuristic algorithms [7]–[12]. Moreover, literature [13] also proposes a multi-objective optimization problem of energy-saving embedding algorithm, which effectively improves the mapping effect. According to the control method of virtual network embedding, we can roughly divide the existing virtual embedding algorithms into distributed embedding algorithms and centralized embedding algorithms.

A. THE DISTRIBUTED EMBEDDING ALGORITHM

Distributed algorithms mainly rely on numerous and widely distributed underlying nodes. The plan of virtual network embedding is completed by each underlying node. Because only when each bottom node has a clear grasp of the global information, it can be possible to make the best solution in the current network environment. For example, the authors of [14] propose a typically distributed algorithm for load balancing, which embeds the virtual network efficiently. For multiple infrastructure providers, literature [12], [15] propose a PolyViNE architecture based on geographic segmentation and embedding to coordinate and resolve the conflict of interest between infrastructure providers and service providers. In addition, while we are providing virtual networks, we must ensure that virtual network services have the capability to respond to sudden failures. Therefore, for the sake of overcoming embedding problem for regional failures, literature [16] proposes a new type of regional disjoint algorithm. This algorithm mainly uses backtracking technology to ensure that the primary VN is disconnected from the backup VN. It also proposes Suurballe algorithm to perfect the embedding between the primary virtual network and the standby virtual network. Experiment shows that this proposed method can debase the chance of underlying resources using up and blocking to a certain extent.

In pace with IoT application scenarios expanding, machine learning and deep learning [17]–[19] have become hot topics in the academic world, especially in the field of smart cities. As the application of machine learning becomes more and more extensive, many scientists innovate virtual network embedding algorithms through reinforcement learning.

In view of the problem that existing heuristic algorithms are easy to converge to local areas, the literature [20] applies deep reinforcement learning(DRL) to the field of VNE. They propose a DRL-VNE algorithm of full attribute matrix (FAM-DRL-VNE). Experiment results have proved that FAM-DRL-VNE algorithm has obvious advantages. The authors of [21] propose a VNE framework (VNE2) based on equivalent bandwidth and guaranteed delay based on current 5G network. Experiments have proved that the proposal of this framework increases the profit of InP in bandwidth allocation. In addition, on the issue of security, the authors of [22] apply blockchain technology to the field of secure VNE and propose a secure SDN problem model. The simulation experiment proves that VNE using blockchain technology has more obvious advantages.

At present, the software defined network can effectively monitor network traffic. At the same time, it introduces a set of new solutions for the rational deployment of virtual network on the problem of network virtualization. Since the inflow capacity of the switch is very important in a virtualized environment, this article [23] proposes an SDN-oriented VN embedding algorithm, which can effectively increase request acceptance rate and revenue of embedding. Similarly, the virtual network embedding algorithm we proposed is also oriented to SDN network architecture. However, the difference is that this article focuses on the consideration of the related problems of finding the optimal solution in the embedding process and realizes a new solution.

B. THE CENTRALIZED EMBEDDING ALGORITHM

The characteristic of the centralized algorithm is that only a single central control entity is responsible for all embedding strategies. The advantage of the centralized algorithm lies in this single central control entity. It collects and evaluates all resource information of the underlying network, so as to grasp various information of the entire network. Therefore, it is more likely to find the optimal or closest embedding scheme to the global optimal. The authors of [9] adopt the two-stage embedding method, the greedy algorithm for the first stage embedding and the shortest path algorithm for the second stage embedding. Similarly, the link mapping phase in [10] also uses the shortest path algorithm. However, the difference is that the algorithm in this article combines geographic location constraints and adds meta-nodes and meta-links to the underlying network to form an augmented topology embed. And finally, it solves the feasible node mapping scheme by using the method of random rounding. The shortest path algorithm is also widely used in solving practical problems [24]. The authors in [25] propose a multi-domain VNE algorithm that minimizes the target overhead. Firstly, a set of candidate panel points conforming to the constraints of virtual embedding is established. Secondly, the feasible ground floor path set is calculated. Finally, Kruskal algorithm is used to embed the path with the minimum weight in the path set, and the corresponding virtual node should be embedded at the same time. At the same time, the authors in [26] also apply reinforcement learning to the virtual embedding problem. They propose a security algorithm by reinforcement learning. The experiment result shows that this intelligent algorithm can get better results in solving virtual embedding problems.

Although the experiment shows that the scheme proposed by literature [9] can meet the basic requirements, there are still some shortcomings in embedding problem of this paper. The literature [27] does not use a two-stage mapping method. It proposes an embedding algorithm based on subgraph isomorphism detection. This method performs node mapping and link mapping at the same time. Experiment results have proved that this method can get better embedding results. However, Baseline algorithm uses a two-stage mapping method. It deploys the controller randomly, uses the greedy algorithm for node mapping, and uses the shortest path method for link mapping. In previous studies on multi-domain virtual network embedding [28]–[30], most scholars used some meta-heuristic algorithms to solve the embedding problem [31]. For example, through the use of neural network algorithm [32], simulated annealing algorithm [33], PSO [34] to refrain from the disadvantage of getting optimal local solution about the greedy algorithm and the shortest path algorithm. The literature [35] uses PSO algorithm for VNE and proposes VNE-PSO algorithm. In this paper, we also use PSO algorithm. However, the difference is that this paper combines PSO and SA algorithms to propose a new virtual embedding scheme, which has better performance than other algorithms. The algorithm proposed in this paper is a centralized algorithm. We manage the information uploaded by each local controller through global controller, and use this information to divide virtual requests.

III. PROBLEM STATEMENT AND NETWORK MODEL

When doing the embedding, service providers need to constantly negotiate with multiple InPs when establishing a multi-domain virtual network request. This paper uses the emerging SDN network architecture, because this architecture can achieve the separation of the forwarding plane and the control plane. It makes that it is possible to clearly recognize the underlying resources and greatly improve the embedding efficiency of VNRs. The multi-domain virtual network architecture includes the following parts:

(1) Infrastructure provider (InP): manage and operate the underlying network.

(2) Service provider (SP): request network resources from InP in the form of virtual network request.

(3) Global controller: receive the virtual network request provided by the SP and divide the virtual network request.

(4) Local controller: upload the underlying data information to the global controller.

A. SUBSTRATE NETWORK MODEL

In a multi-domain physical network, the data domain can be represented by a power undirected graph $G_s = \{\{G_s^i\}, C_s^n, C_s^l\}$. $G_s^i = \{N_s^i, L_s^i\}$ in the expression means the i-th domain in the network. $N_s^i = \{n_s^{i1}, n_s^{i2}, n_s^{i3} \ldots\}$ means total nodes number of the i-th domain. And L_s^i represents a collection of physical network links in the i-th domain. C_s^n means available computing power of the underlying node, and C_s^l represents the remaining available bandwidth resource of the underlying link.

As shown in FIGURE 2, it depicts an underlying network. There are a total of nine substrate nodes in the underlying network model. The number of the orange rectangular frame nearby the node represents available computing resources of the underlying node. The substrate network includes eleven substrate links in total, and the number nearby every substrate link represents corresponding available bandwidth resources.

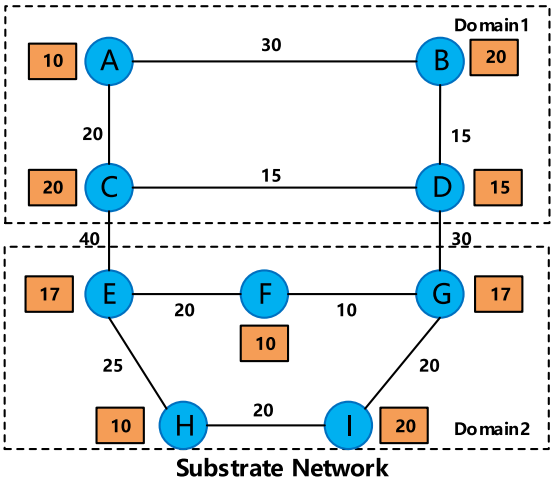


FIGURE 2. Substrate network.

B. VIRTUAL NETWORK REQUEST MODEL

Similarly, we use an authorized undirected graph $G_v = \{N_v, L_v, C_v^n, C_v^l\}$ to represent the virtual network model. $N_v = \{n_v^1, n_v^2, n_v^3 \ldots\}$ means total number of virtual nodes and n_v^i represents the i-th virtual node. C_v^n represents the computing demand of virtual node n_v, that is, the CPU demand $CPU\ (n_v)$. C_v^l means bandwidth resource demand $BW(l_v)$ about link l_v.

As shown in FIGURE 3, it depicts a virtual network request. It contains 4 virtual nodes {a, b, c, d}, and 4 virtual links {a-b, b-c, c-d, a-d}. The number of the box next to the node means available CPU resources. The number next to virtual link means bandwidth resources of virtual links.

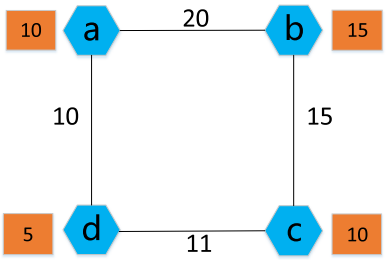


FIGURE 3. Virtual network request.

C. VIRTUAL NETWORK EMBEDDING PROBLEM STATEMENT

The underlying basic network of the virtual embedding is managed by multiple InPs. Different InPs manage their respective nodes and links, and the information between each InP is not disclosed. Because candidate domains embedded by virtual nodes are different, the VRs will pass through domains managed by different InPs. When performing multi-domain embedding, the virtual network needs to be distributed to different domains, and then the respective InP completes the corresponding intra-domain embedding. The virtual network embedding problem can be defined as an embedding that must meet the embedding constraints from VNR topology G_v to the underlying network topology G_s: $G_v\{N_v, L_v\} \rightarrow G_s\left\{N_s', L_s'\right\}$, where $N_s' \in N_s, L_s' \in L_s$. In the node mapping stage, virtual node is embedded to underlying node that meets its resource constraints. In the link mapping stage, virtual link is embedded to the underlying acyclic path that meets its bandwidth resource constraints.

(1) node mapping: The goal of node mapping is to find a suitable physical node in the underlying network for embedding, that is, CPU resources of selected node n_s^i $\left(n_s^i \in N_s\right)$ need to meet the constraints of the virtual node n_v^i.

$$CPU\left(n_s^i\right) \geq CPU\left(n_v^i\right), \tag{1}$$

$$n_s^i \in CD\left(n_v^i\right). \tag{2}$$

Formulas (1) and (2) indicate the constraints on the substrate node to ensure that the residual resource amount can satisfy the requirement of the virtual node. Formula (1) indicates that the CPU resource amount of the underlying physical node $\left(n_s^i\right)$ must be greater than or equal to the resource demand of the node $\left(n_v^i\right)$. Formula (2) indicates that the underlying node $\left(n_s^i\right)$ must be located in the candidate physical domain of virtual node $\left(n_v^i\right)$. $CD\left(n_v^i\right)$ represents the candidate domain for the virtual node. That is, in the embedding process, it can only be embedded in the specified physical domain. Only if both of these conditions are satisfied, the virtual node can be successfully embedded.

(2) link mapping: Link mapping is divided into inter-domain link mapping and intra-domain link mapping. Firstly, the local controller is responsible for the relevant intra-domain link mapping. Secondly, the global controller performs inter-domain mapping according to the information conveyed by the local controller. At any stage, the virtual link needs to select a physical link that meets the embedding conditions for embedding. That is, bandwidth resource of physical link $\left(n_s^i, n_s^j\right)$ needs to meet the virtual link $\left(n_v^u, n_v^v\right)$ bandwidth resources requirements. Only when the following two formulas are met, the virtual link can be successfully embedded.

$$BW\left(n_s^i, n_s^j\right) \geq BW\left(n_v^u, n_v^v\right), \tag{3}$$

$$\sum BW\left(L_i^j\right) \geq \sum BW\left(L_u^v\right). \tag{4}$$

Among them, formula (3) indicates that bandwidth resource on the underlying link needs to satisfy bandwidth resource demands, that is, bandwidth resources of physical link $\left(n_s^i, n_s^j\right)$ must be greater than or equal to bandwidth resources demand of the road $\left(n_v^u, n_v^v\right)$. Formula (4) means that total bandwidth resources about underlying network should be greater than or same with total bandwidth resources demand about virtual links. When these two formulas are satisfied, the virtual link can be successfully embedded.

FIGURE 4 depicts an embedding example of VNR embedded to the underlying network. Among them, the node mapping scheme is $\{a \rightarrow A, b \rightarrow B, c \rightarrow D, d \rightarrow C\}$, and the link mapping scheme is $\{(a, b) \rightarrow (A, B), (b, c) \rightarrow$

(B, D) , $(c, d) \rightarrow (D, C)$, $(a, d) \rightarrow (A, C)\}$. Therefore, virtual requests can be successfully allocated under the demands of virtual node capacity and link bandwidth.

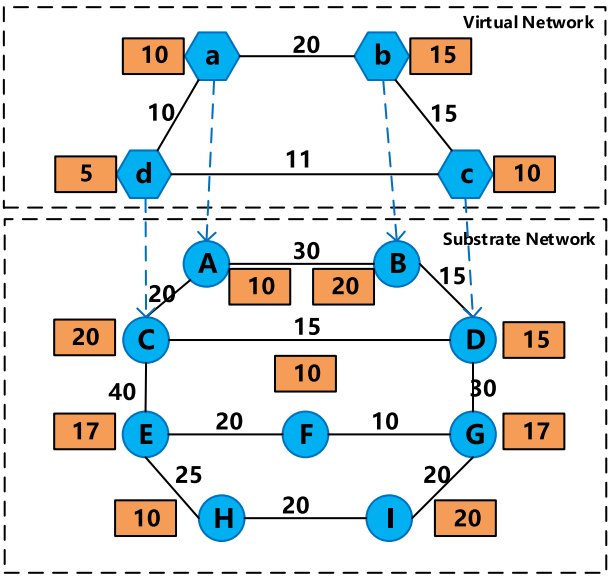


FIGURE 4. Virtual network embedding.

D. VIRTUAL NETWORK EMBEDDING EVALUATION INDEX

Our main work is to make some innovations in the virtual network embedding problem, combining with the actual network conditions, trying to optimize the embedding overhead and improve the request acceptance rate. Therefore, to comprehensively assess the performance of the hybrid particle swarm algorithm, this paper takes virtual embedding overhead, virtual request successful acceptance rate, and the embedding revenue overhead ratio as the main evaluation indicators.

The embedding benefit for time t is defined as the sum of computing resources and bandwidth resources required by the virtual request.

$$Revenue\,(G_v, t) = \sum_{n_v \in N_v} CPU(n_v) + \sum_{l_v \in L_v} BW(l_v). \quad (5)$$

The embedding overhead for time t is defined as the sum of physical resources by underlying network.

$$Cost\,(G_v, t) = \sum_{n_v \in N_v} CPU + \sum_{l_v \in L_v} \sum_{l_s \in L_s} BW(f_{l_s}^{l_v}, l_v). \quad (6)$$

Formula (6) represents the embedding overhead of a virtual request. Among them, CPU refers to the CPU resources allocated to virtual nodes by the underlying network, and $BW(f_{l_s}^{l_v}, l_v)$ refers to the bandwidth resources allocated to virtual links by the underlying network. $f_{l_s}^{l_v}$ is a boolean variable, $f_{l_s}^{l_v} \in (0, 1)$, which represents the embedding relationship of the link. When the underlying link l_s allocates bandwidth resources to the virtual link l_v, $f_{l_s}^{l_v} = 1$; otherwise, $f_{l_s}^{l_v} = 0$.

Therefore, the main evaluation indicators of virtual embedding are as follows:

(1) The embedding cost of the underlying network:

$$\lim_{T \to \infty} \frac{\sum_{t=0}^{T} Cost(G_v, t)}{T}. \quad (7)$$

(2) The request acceptance rate:

$$\lim_{T \to \infty} \frac{\sum_{t=0}^{T} VNR_{accept}}{\sum_{t=0}^{T} VNR_{arrive}}. \quad (8)$$

The request acceptance rate means the percentage of VNRs embedded successfully over a period of time as a percentage of the total virtual network requests. $\sum_{t=0}^{T} VNR_{accept}$ refers to the quantity of VNRs embedded successfully in that time, and $\sum_{t=0}^{T} VNR_{arrive}$ means the total requests in that time. When the ratio of requests accepted is higher, it indicates the function of embedding scheme is better.

(3) The long-term embedding revenue overhead ratio of the underlying network:

$$\lim_{T \to \infty} \frac{\sum_{t=0}^{T} Revenue(G_v, t)}{\sum_{t=0}^{T} Cost(G_v, t)}, \quad (9)$$

the revenue overhead ratio reflects the efficiency in the use of underlying resources. When the ratio is larger, the utilization of physical resources is larger, so that the result of embedding algorithm is better.

IV. VNE MODEL

Considering to lower the embedding cost and improve the embedding efficiency, an integer linear programming model (ILP) is established to obtain the optimal solution.

A. OBJECTIVE FUNCTION

In the same virtual request, the CPU cost for different solutions is the same, but the consumption is different in terms of bandwidth due to the different bandwidth path selections, resulting in different underlying network consumption. Therefore, the objective function is represented by the following formula:

$$Minisize \sum_{(u,v) \in L_v} \sum_{(i,j) \in L_s} f_{ij}^{uv} \times BW(l_{uv}), \quad (10)$$

among them, the binary variable f_{ij}^{uv} means that the variable takes the value 1 if the virtual link (u, v) is successfully embedded to (i, j). Otherwise it takes the value 0. $(u, v) \in L_v$ represents the virtual link (u, v) in the virtual network, and $(i, j) \in L_s$ represents the physical link (i, j) in the underlying network. For virtual network requests, reducing embedding overhead can improve resource utilization and network performance. Therefore, formula (10) is used as the objective function. This objective function minimizes the request embedding overhead.

B. NODE CONSTRAINTS

When embedding virtual nodes, two conditions are met to embed successfully. One is that resources of underlying nodes must conform to resource requirements of virtual nodes.

Second, underlying physical node must be in the candidate domain CD(u) of this node u.

$$\begin{cases} \forall u \in N_v, \forall i \in N_s, \\ x_i^u \times CPU(u) \leq CPU(i), \\ x_i^u \times i \leq CD(u), \end{cases} \tag{11}$$

the variable x_i^u in formula (11) is a binary variable. It means when the embedding between the physical node i and virtual node u is successful, this variable value is 1. Otherwise the value is 0.

C. LINK CONSTRAINTS

During link mapping, the bandwidth resources of physical link should be guaranteed to correspond to the sum required by virtual link and the control channel embedded on it, as shown in equation (12).

$$\forall (i,j) \in L_s, \forall (u,v) \in L_v, f_{ij}^{uv} \times BW(l_{uv}) \leq BW(l_{ij}). \tag{12}$$

If the nodes u and v are embedded to the underlying network nodes i and j respectively, virtual link (u, v) will be embedded to an underlying path P from i to j in link mapping period. The equation (13) expresses the connectivity constraints of virtual links.

$$\forall i,j \in N_s, \forall (u,v) \in L_v,$$
$$\sum_{(i,j)\in L_s} f_{ij}^{uv} - \sum_{(j,i)\in L_s} f_{ji}^{uv} = \begin{cases} 1, & x_i^u = 1 \\ -1, & x_i^v = 1 \\ 0, & otherwise \end{cases} \tag{13}$$

when the value of formula (13) is 1, it means that if a physical link contains the source node of the virtual link, the outgoing flow of source node i is 1, and the incoming flow is 0. When the value is -1, it means that if a physical link contains the destination node of the virtual link, the incoming flow of destination node j is 1, and the outgoing flow is 0. When the value is 0, it means that if a physical link does not include source node and destination node, the outgoing and incoming traffic is balanced.

D. VARIABLE CONSTRAINTS

For the same VNR, the underlying node cannot be repeatedly embedded by multiple virtual nodes. It can only carry one virtual node. Therefore, it has the following constraints:

$$\begin{aligned} \forall i \in N_s, \sum_{\forall u \in N_v} x_i^u \leq 1, \\ \forall u \in N_v, \sum_{\forall i \in N_s} x_i^u = 1, \\ \forall u \in N_v, \quad \forall i \in N_s, x_i^u \in \{0,1\}, \\ \forall (u,v) \in L_v, \quad \forall (i,j) \in L_s, f_{ij}^{uv} \in \{0,1\}. \end{aligned} \tag{14}$$

V. ALGORITHM IMPLEMENTATION

The mathematical model for solving the VNE is an NP-hard problem. PSO algorithm is an intelligent group optimization algorithm based on population. At present, PSO is widely used in the fields of engineering, economy and science. In this paper, the PSO method is introduced into the VNE problem. For the sake of solving VNE problem more effectively, PSO has been optimized to make this algorithm more suitable for the embedding problem. It can guarantee the global optimality of the optimization results. Therefore, this section designs VNE-HPSO embedding algorithm based on PSO to solve the embedding problem.

A. PSO ALGORITHM THEORY BASIS

The particle swarm algorithm originated from the study of bird predation behavior. Similar to other evolutionary algorithms, PSO algorithm can guide the whole group to the direction of possible solutions through information transmission among individuals, and find better solutions in the process. In the particle swarm algorithm, each particle has a position variable and a velocity variable. Their position represents a solution to the problem. When solving the problem, the particles move to the optimal solution at a certain speed. Through a limited number of iterations, the historical best position is continuously updated. Finally, a better particle position, that is, the solution to the problem, is obtained. During this period, the particles update their velocity and position according to formula (15) and formula (16).

$$V_{i+1} = wV_i + c_1 r_1 (X_{pb} - X_i) + c_2 r_2 (X_{gb} - X_i), \tag{15}$$
$$X_{i+1} = X_i + V_{i+1}, \tag{16}$$

among them, the vector V_i represents the speed at this time, which represents the tendency of the particle to search for a new area. The updated velocity and position on the basis of equation (15) and equation (16) are the vector V_{i+1} and the vector X_{i+1}. X_i means the location at this time; w represents the weight inertia of the particle maintaining position information; c1 and c2 represents the tendency to move towards the optimal solution; r1 and r2 are a random number between (0,1); X_{pb} represents the historical best location of the particle, and X_{gb} means the historical best location of all particles.

B. REDEFINITION OF RELATED PARAMETERS

On the basis of our proposed virtual network embedding model, the relevant parameters are redefined as follows:

1) Particle current position: The particle position vector $X_i = \left(x_i^1, x_i^2, \ldots, x_i^n\right)$ represents the i-th embedding scheme. The vector x_i^k represents the physical node number embedded by the virtual node n_k.

2) Particle current speed: The current speed of the particle $V_i = \left(v_i^1, v_i^2, \ldots, v_i^n\right)$ indicates the adjustment of the embedding scheme, so that the current scheme is adjusted to the optimal embedding scheme gradually. Each dimension in the vector is a binary variable with a value of 0 or 1.

3) $\ominus$ calculation: $X_i \ominus X_j$ represents the difference between two vectors, that is, the difference between the two embedding schemes. If the value of the vector X_i and the vector

X_j is same, the difference result is 1. Otherwise, it is 0. For example, $(1, 3, 4, 6, 2) \ominus (2, 3, 5, 6, 9) = (0, 1, 0, 1, 0)$.

4) $\oplus$ calculation: $pV_i \oplus qV_j$ represents the adjustment decision of the embedding scheme. Among them, p and q represent the probability respectively. pV_i means that each dimension in the vector V_i takes a value with the probability of p, qV_j means that each dimension in the vector V_j takes a value with the probability of q, and $p + q = 1$. For example, $0.4\,(1, 0, 0, 1, 1) \oplus 0.6\,(1, 0, 1, 1, 0) = (1, 0, *, 1, *)$, where * means that the value of the dimension is uncertain about 0 or 1. Among them, the first * indicates that the value of this dimension takes 0 with a probability of 0.4 and 1 with a probability of 0.6.

5) $\otimes$ calculation: $X_i \otimes V_j$ means obtaining new embedding plan. During calculation, embedding plan X_i adjusts the embedding node on the basis of the policy V_j. For example, $(1, 2, 5, 4) \otimes (1, 0, 1, 1) = (1, 0, 5, 4)$, it means that the second node ought to change its embedding plan.

Therefore, the position and velocity formulas in the redefined PSO algorithm are adjusted as follows:

$$V_{i+1} = \alpha V_i \oplus \beta(X_{pb} \ominus X_i) \oplus \gamma(X_{gb} \ominus X_i), \quad (17)$$

$$X_{i+1} = X_i \otimes V_{i+1}, \quad (18)$$

among them, for the inertia weight α, we adopt a linear differential decline strategy. It can better balance the search ability during the iteration.

C. SA ALGORITHM

The idea of simulated annealing (SA) algorithm was first proposed by N. Metropois *et al.* in 1953. The simulated annealing method is a random search method that simulates the metal annealing process. The advantages of the SA algorithm are its wide range of application, high reliability for obtaining the optimal global solution, simple algorithm, and easy implementation. The main difference between the SA algorithm and the traditional random search method in search strategy is that it not only introduces suitable random factors, but also introduces the natural mechanism of annealing and cooling process in physical systems. Compared with the particle swarm algorithm that only replaces the original position with a better value, the physical annealing mechanism introduced by the SA algorithm makes it accepts the new state with probability. In other words, if the current state is worse than the original state, it may happen that the current state replaces the original state, and the probability of acceptance decreases as the temperature drops. Therefore, SA algorithm has a mechanism to get rid of local optimum. It can effectively avoid the possibility of obtaining the local optimum solution during the search process, and increase the feasibility of finding the global optimum.

D. THE ALLOCATION STRATEGY OF PARTICLE INITIALIZATION

In this section, we elaborate on the particle initialization allocation strategy. It mainly describes how to allocate particles. In the traditional PSO algorithm, initializing the particles is to randomly generate the location and speed with the same probability. This will cause network fragmentation and waste of resources. However, the particle initialization allocation strategy can reduce the generation of underlying network fragments and improve resource utilization. In addition, it reduces unnecessary iterative processes in the later stages of the algorithm and improves iterative efficiency.

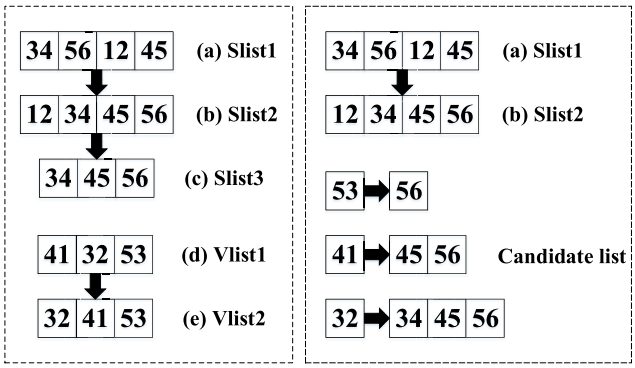


FIGURE 5. Comparison between the traditional random initialization of particles and the allocation strategy of particle initialization.

Under normal conditions, we arrange the physical nodes and virtual nodes in descending order of the number of resources firstly, and find the lowest resource request amount among the virtual nodes. Then we abandon the physical nodes that are smaller than its resource amount from the physical node list. To begin with, a physical node is allocated to every virtual node, but the allocated node may not be able to meet the requested amount of resources. In FIGURE 5, the list of node resources requested is {41, 32, 73}, and the list of resources of physical nodes is {45, 34, 56, 12}. We sort them in descending order to get $Vlist2 = \{32, 41, 73\}$, $Slist2 = \{12, 34, 45, 56\}$. In the physical node list, we can get $Slist2 = \{34, 45, 56\}$ after going to the physical node less than the minimum requested resource amount. For each node in the virtual node list, the probability of success of the assigned physical node is 1/3, 1/2, and 1 respectively. Therefore, the particle initialization process can be further optimized to reduce some unnecessary iterative processes. For particle initial allocation policy, as with traditional particle allocation, we first rank the virtual network request resources and physical network resources in descending order. Nodes and links that are less than the minimum resource requests are then removed from the physical node list. Secondly, all virtual nodes are assigned a candidate node list, and the number of candidate node resources in this candidate node list must be no less than the minimum amount of requested resources for this virtual node. Finally, the probability of success of virtual node allocation is 1,1,1 respectively. Thus it can be seen that the allocated candidate node list meets the resources required through this strategy. The particle initialization distribution strategy greatly improves the probability of successful particle

distribution, thereby reducing the time cost and accelerating the convergence speed of the virtual network embedding algorithm.

In time complexity analysis, we set the number of virtual nodes as N_v and the number of physical nodes as N_s. We use the quick sort method to sort the virtual node sequence and the physical node sequence respectively, the complexity of which is $O(\log N_v)$ and $O(\log N_s)$ respectively. Then assign candidate nodes to each virtual node, and the complexity is $O(N_v \cdot N_s)$.

E. VNE-HPSO ALGORITHM

We give the embedding procedure of VNE-HPSO algorithm in Algorithm 1.

Algorithm 1 The Proposed VNE-HPSO Algorithm

Require: Virtual Network Request $G_v = \{N_v, L_v\}$, Substrate Network Topology $G_s = \{N_s, L_s\}$, *iteration*
Ensure: VNE gBest

```
VNetwork vn, SNetwork sn;
Iteration Number iteration;
count=0;
Initialize particle position;
Maximum fitness f_max, Minimum fitness f_min;
Initial temperature t_0= f_max − f_min;
Particle current position pBest, the best particle position gBest;
for all the unmapped virtual links in VNR do
    for count < iteration do
        update the location and velocity;
        if the fitness is superior to the value of pBest then
            set pBest to the new location;
        end if
        if the fitness is superior to the value of gBest then
            set gBest to the new location;
        end if
        randomly generate a new position for the particle, and calculate the fitness difference between the new and the old position ΔC;
        if ΔC < 0 then
            the particle enters new position;
        end if
        produce a number r ∈ (0, 1) at random;
        if r < min[1, exp(−ΔC/t)] then
            the particle enters new position and update pBest, gBest;
        end if
        t=t*0.9;
        count++;
    end for
    return gBest;
end for
```

F. TIME COMPLEXITIES ANALYSIS

We set the number of virtual request nodes as N_v, the number of physical nodes as N_s, the number of data fields as D, the number of iterations as L, and the number of particles as C. Then the complexity analysis of the whole algorithm is as follows.

$$O(L \cdot C \cdot N_v \cdot D \cdot N_v \cdot D \cdot N_s) = O(L \cdot C \cdot N_s \cdot D^2 \cdot N_v^2). \tag{19}$$

where, C, L, D, N_s are all fixed values, so the complexity of the whole algorithm can be regarded as $O(N_v^2)$. At the same level as other heuristic algorithms.

VI. SIMULATION EXPERIMENTS AND ANALYSIS

For the sake of verifying the performance of VNE-HPSO algorithm, we compare it with the previous algorithm through simulation experiments. First of all, this paper introduces the environment of the simulation experiment and some parameters. Secondly, we evaluate the performance of the algorithm on account of the main evaluation indicators given in this article.

A. EXPERIMENTAL ENVIRONMENT AND PARAMETER SETTINGS

This simulation experiment uses Java as the programming language. And in the experiment environment, the computer we used is 8GB memory and carries an x64-based Intel(R) Core(TM) i5-6200U CPU processor. We use GT-ITM [36] tools to randomly generate underlying network and virtual network embedding requests required for the experiment. For contrasting the performance of this virtual network embedding algorithm, we set the relevant parameters in the simulation experiment, as shown in TABLE 1.

TABLE 1. Parameter setting.

Parameter	The Value
physical nodes	50
the initial resources of the nodes	U[100, 300]
the cost of the nodes in the domain	U[1 , 10]
the intra-domain node connection rate	50%
initial resources of the link	U[1000,3000]
bandwidth price of the inter-domain link	U[5,15]
requested resources of virtual nodes	U[1,10]
intra-domain node connection rate	50%
the number of particles	10
the number of iterations	50
maximum inertia weight	0.9
minimum inertial weight	0.4
β	0.3
γ	0.4

B. EXPERIMENTAL ANALYSIS

Since this article is mainly aimed at improving PSO algorithm, a typical VNE-PSO algorithm is selected for comparison. In addition, we have selected the basic two-stage embedding algorithm Baseline and the one-stage embedding algorithm D-ViNE-SP respectively. This article compares VNE-HPSO with VNE-PSO, D-ViNE-SP and Baseline algorithms, and verifies the effectiveness of the VNE-HPSO algorithm according to the evaluation index proposed by formulas (7) (8) (9). For the request acceptance rate, we calculate the number of successfully embedded requests and

the total number of requests issued by the virtual network respectively. And it is obtained by calculating their ratio. For the revenue cost ratio, it is obtained by calculating the ratio of embedding revenue and embedding cost, where the embedding revenue and embedding cost are calculated by formula (5) and (6). For the running time, we set the initial time and the end time respectively in the experiment. It is obtained by calculating the difference between them. The following are simulation experiments for these aspects.

Experiment 1: Comparison of embedding costs in different virtual nodes

For the sake of contrasting the performance of different algorithms in terms of embedding overhead, we conduct a simulation experiment. The experimental results are shown in FIGURE 6 and FIGURE 7. The difference between the two graphs is the set of independent variables. FIGURE 6 uses the number of virtual nodes as an independent variable to observe embedding overhead. FIGURE 7 uses the number of data fields as the independent variable. The results of FIGURE 6 show the changes in virtual network embedding overhead corresponding to four different algorithms under the premise that the data field is 4. Among them, the VNE-HPSO algorithm has the least cost in the embedding process. There is an obvious contrast with remaining algorithms. It outperforms in terms of embedding overhead. Through quantitative analysis of the data, the VNE-HPSO algorithm is 32.82% lower than VNE-PSO in terms of embedding overhead, 39.3% lower than D-ViNE-SP, and 50.47% lower than Baseline.

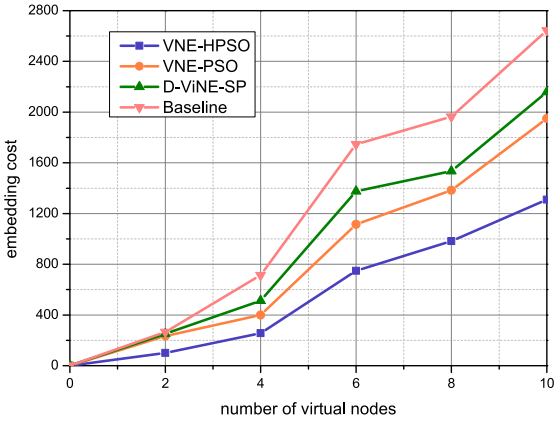


FIGURE 6. Comparison of costs of four algorithms under 4 data domains.

Experiment 2: Comparison of embedding costs in different data domains

Not only that, as shown in FIGURE 7, we also analyze the changes in embedding overhead with different data domains under the condition of a certain virtual request scale. When there are 6 virtual nodes, we analyze the embedding overhead when the data fields are 4, 6, and 8, respectively. From the resultant picture, we can find that the VNE-HPSO algorithm always performs best and can

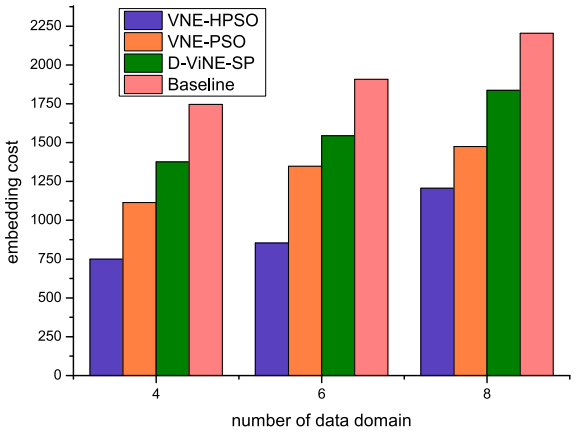


FIGURE 7. Comparison of costs of four algorithms under 6 virtual nodes.

give the least costly embedding scheme. Compared with the other three schemes, this scheme is 32.7% lower than VNE-PSO, 45.5% lower than D-ViNE-SP, and 57.1% lower than Baseline.

Experiment 3: Comparison of VR acceptance rate

Simulation results about VNR acceptance rate of embedding algorithms are shown in FIGURE 8. From the resultant picture, we can find that under the same experimental environment, as time goes on, the VR acceptance rate about embedding algorithms gradually cuts down and finally stabilizes. Moreover, the resultant picture shows that the ratio of VNE-HPSO algorithm is the highest, and the ratio of Baseline is the lowest. This is because at the beginning of VNE-HPSO algorithm, we introduce particle initialization distribution strategy, which is not easy to generate network fragments when randomly assigning particle positions. This strategy provides a greater possibility to receive the virtual network. While Baseline chooses to embed the node with the most remaining resources during the embedding process, the available physical resources may not be able to meet the subsequent virtual requests with high resource requirements in the later stage, resulting in embedding failure. Through comparison with other three algorithms, the VR

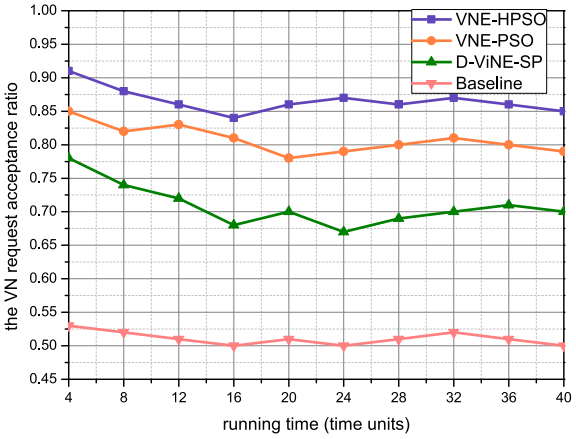


FIGURE 8. Comparison of virtual request acceptance rate of four algorithms.

acceptance rate of the VNE-HPSO algorithm is 7% higher than VNE-PSO, 17.6% higher than D-ViNE-SP, and 41.2% higher than Baseline.

Experiment 4: Comparison of revenue cost ratio of four algorithms

In FIGURE 9, we compare the revenue cost ratios of the four algorithms in the embedding process. From the resultant picture, we can find that the revenue cost ratios about four embedding algorithms have not changed much and are relatively flat. The VNE-HPSO algorithm has the highest final ratio in terms of revenue cost ratio, which is stable at around 68%. It is precise because of our hybrid particle swarm algorithm that we can obtain the optimal global solution more accurately, thereby reducing the virtual network embedding overhead. Therefore, the VNE-HPSO algorithm has the highest ratio.

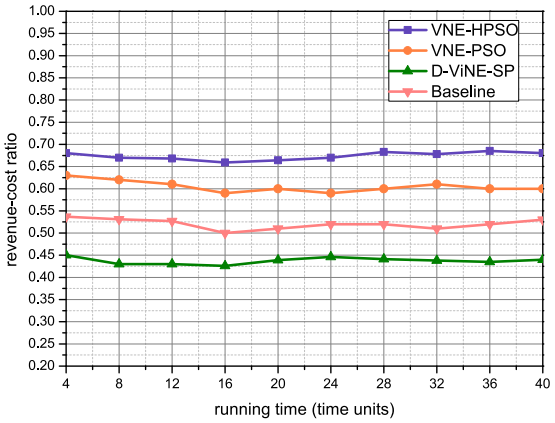


FIGURE 9. Comparison of revenue-cost ratio of four algorithms.

Experiment 5: Comparison of running time of four algorithms

FIGURE 10 shows the results about the running time of four embedding schemes. Obviously, as shown in the figure, the VNE-HPSO algorithm has the lowest running time. Through the collation of data and analysis, the VNE-HPSO algorithm is 18.7% lower than VNE-PSO in the way of running time, 50% lower than D-ViNE-SP, and 56.6% lower than Baseline.

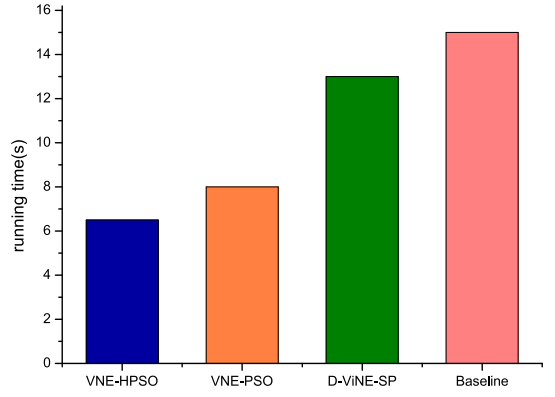


FIGURE 10. Comparison of running time of four algorithms.

From the above resultant pictures, we can find that the VNE-HPSO algorithm is superior to VNE-PSO, D-ViNE-SP, and Baseline in terms of embedding overhead, virtual request acceptance rate and running time. Moreover, as time goes by, the performance of VNE-HPSO algorithm is within a controllable range. In the algorithm design of Baseline, node mapping is performed first through the greedy algorithm, and then the shortest path algorithm is used for link mapping. During node mapping, adjacent nodes may be embedded to nodes farther apart, which seriously increases the cost of link mapping. Next, the problem of insufficient resources may occur at the end of algorithm, which also leads to a low VR acceptance rate.

Both VNE-PSO and D-ViNE-SP algorithm perform virtual embedding when underlying link does not support breaking up, but the result of VNE-PSO algorithm stays ahead of the D-ViNE-SP algorithm. That is because the ultimate solution received by D-ViNE-SP is not an optimal solution to a certain extent, or even a feasible solution. Through the VNE-PSO algorithm, we can get the approximate global optimal solution.

Similarly, compared with VNE-PSO, the VNE-HPSO algorithm does not support the underlying link splitting. However, due to the introduction of the particle initialization allocation strategy, the use of underlying resources is more efficient, the generated network fragments are reduced, and the VR acceptance rate is enhanced. At the same time, the annealing process is introduced into the PSO algorithm, which avoids the problem of premature convergence in the course of finding optimal solution, and makes the algorithm performance more efficient.

VII. CONCLUSION

For the sake of solving the VNE problem more effectively, this paper proposes an embedding programme based on hybrid particle swarm. The VNE-HPSO algorithm, particle initialization distribution strategy and linear differential reduction strategy of inertia weight proposed in this paper strikingly optimize the property about VNE, and play an important role in reducing the embedding overhead and improving the VR acceptance rate. Experimental results prove the effectiveness of this method. In addition, the algorithm proposed in this paper is of great significance in the fields of tracking objects and data mining. For the linear differential decline strategy, we can use its convergence performance to apply it to fields such as optimal scheduling.

But for one thing, in actual multi-domain VN environment, InPs is unwilling to announce all information about resource utilization to other parties. Consequently, the function of the algorithm is unable to be tested in a better way. For another, the mathematical foundation of the particle swarm algorithm is relatively weak, lacking comprehensive and profound theoretical analysis. These are all directions that we need to focus on in our future work.

REFERENCES

[1] P. Zhang, C. Jiang, X. Pang, and Y. Qian, "STEC-IoT: A security tactic by virtualizing edge computing on IoT," *IEEE Internet Things J.*, early access, Aug. 19, 2020, doi: 10.1109/JIOT.2020.3017742.

[2] P. Zhang, H. Yao, and Y. Liu, "Virtual network embedding based on the degree and clustering coefficient information," *IEEE Access*, vol. 4, pp. 8572–8580, 2016.

[3] P. Zhang, H. Yao, and Y. Liu, "Virtual network embedding based on computing, network, and storage resource constraints," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3298–3304, Oct. 2018.

[4] P. Zhang, H. Yao, C. Qiu, and Y. Liu, "Virtual network embedding using node multiple metrics based on simplified ELECTRE method," *IEEE Access*, vol. 6, pp. 37314–37327, 2018.

[5] D. Thakur and M. Khatua, "Multi-domain virtual network embedding with dynamic flow migration in software-defined networks," *J. Netw. Comput. Appl.*, vol. 162, Jul. 2020, Art. no. 102639.

[6] C. Wang and T. Wolf, "Virtual network mapping with traffic matrices," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Oct. 2011, pp. 225–226.

[7] Y. Hongfang, C. Qiao, V. Anand, X. Liu, H. Di, and G. Sun, "Survivable virtual infrastructure mapping in a federated computing and networking system under single regional failures," in *Proc. GLOBECOM*, Dec. 2010, pp. 1–6.

[8] Y. Hongfang, V. Anand, C. Qiao, and G. Sun, "Cost efficient design of survivable virtual infrastructure to recover from facility node failures," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2011, pp. 1–6.

[9] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: Substrate support for path splitting and migration," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 17–29, Mar. 2008.

[10] N. M. M. K. Chowdhury, M. R. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," in *Proc. 28th Conf. Comput. Commun. (INFOCOM)*, Apr. 2009, pp. 783–791.

[11] M. Feng, J. Liao, J. Wang, S. Qing, and Q. Qi, "Topology-aware virtual network embedding based on multiple characteristics," in *Proc. IEEE Int. Conf. Commun. (ICC)*, vol. 8, Jun. 2014, pp. 2956–2962.

[12] F. Samuel, M. Chowdhury, and R. Boutaba, "PolyViNE: Policy-based virtual network embedding across multiple domains," *J. Internet Services Appl.*, vol. 4, no. 1, p. 6, Sep. 2013.

[13] A. Jahani, L. M. Khanli, M. T. Hagh, and M. A. Badamchizadeh, "EE-CTA: Energy efficient, concurrent and topology-aware virtual network embedding as a multi-objective optimization problem," *Comput. Standards Interfaces*, vol. 66, Oct. 2019, Art. no. 103351.

[14] I. Houidi, W. Louati, and D. Zeghlache, "A distributed virtual network mapping algorithm," in *Proc. IEEE Int. Conf. Commun.*, pp. 5634–5640, Jun. 2008.

[15] J. Wang, J. Liao, M. Feng, L. Zhang, and Q. Qi, "Topology-aware virtual network embedding through the degree," in *Proc. Nat. Doctoral Acad. Forum Inf. Commun. Technol.*, Jan. 2013, p. 16.

[16] C. Galdamez and Z. Ye, "Resilient virtual network mapping against large-scale regional failures," in *Proc. Wireless Opt. Commun. Conf.*, Apr. 2017, pp. 1–4.

[17] C. Jiang and Z. Li, "Decreasing big data application latency in satellite link by caching and peer selection," *IEEE Trans. Netw. Sci. Eng.*, early access, May 14, 2020, doi: 10.1109/TNSE.2020.2994638.

[18] C. Jiang, H. Zhang, Y. Ren, Z. Han, K.-C. Chen, and L. Hanzo, "Machine learning paradigms for next-generation wireless networks," *IEEE Wireless Commun.*, vol. 24, no. 2, pp. 98–105, Apr. 2017.

[19] C. Jiang and X. Zhu, "Reinforcement learning based capacity management in multi-layer satellite networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 7, pp. 4685–4699, Jul. 2020.

[20] P. Han, L. Guo, and Y. Liu, "VNE2: A virtual network embedding framework based on equivalent bandwidth in fiber-wireless enhanced 5G networks," in *Proc. 22nd Int. Conf. Transparent Opt. Netw. (ICTON)*, Jul. 2020, pp. 1–4.

[21] S. Zhang, C. Wang, J. Zhang, Y. Duan, X. You, and P. Zhang, "Network resource allocation strategy based on deep reinforcement learning," *IEEE Open J. Comput. Soc.*, vol. 1, pp. 86–94, Jun. 2020.

[22] C. Haotong, Y. Hu, Q. Wang, S. Wu, and L. Yang, "A blockchain-based virtual network embedding algorithm for secure software defined networking," in *Proc. IEEE Conf. Comput. Commun. Workshops*, Jul. 2020, pp. 1057–1062.

[23] H. Kuang and Y. Fu, "VNE-TS: A novel virtual network mapping algorithm in SDN," in *Proc. 5th Int. Conf. Comput. Sci. Netw. Technol.*, Dec. 2016, pp. 657–661.

[24] P. Zhang, X. Pang, Y. Bi, H. Yao, H. Pan, and N. Kumar, "DSCD: Delay sensitive cross-domain virtual network embedding algorithm," *IEEE Trans. Netw. Sci. Eng.*, early access, Jul. 10, 2020, doi: 10.1109/TNSE.2020.3005570.

[25] L. Peng, "A multi-domain virtual network embedding algorithm based on minimum cost," *J. South China Univ. Technol. (Natural Sci. Ed.)*, vol. 43, pp. 67–73 and 112, Sep. 2015, doi: 10.3969/j.issn.1000-565X.2015.09.011.

[26] P. Zhang, C. Wang, C. Jiang, and A. Benslimane, "Security-aware virtual network embedding algorithm based on reinforcement learning," *IEEE Trans. Netw. Sci. Eng.*, early access, May 19, 2020, doi: 10.1109/TNSE.2020.2995863.

[27] J. Lischka and H. Karl, "A virtual network mapping algorithm based on subgraph isomorphism detection," in *Proc. 1st ACM Workshop Virtualized Infrastruct. Syst. Archit.*, Aug. 2009, pp. 81–88.

[28] M. Jiang, Z. Zhao, M. Zhang, J. Tang, C. Wu, and X. Min, "A virtual network mapping algorithm based on time," *Chin. J. Electron.*, vol. 23, pp. 31–36, Jan. 2014.

[29] Q. Xu, H. Yi, J. Zhu, and H. Hu, "Virtual network mapping algorithm based on entropy weight method," *Comput. Eng. Appl.*, vol. 51, p. 94, Jan. 2015.

[30] S. Li, M. Y. Saidi, and K. Chen, "Multi-domain virtual network embedding with coordinated link mapping," in *Proc. 24th Int. Conf. Softw., Telecommun. Comput. Netw. (SoftCOM)*, Sep. 2016, pp. 1–6.

[31] P. Zhang, H. Yao, M. Li, and Y. Liu, "Virtual network embedding based on modified genetic algorithm," *Peer Peer Netw. Appl.*, vol. 12, no. 2, pp. 481–492, Oct. 2017.

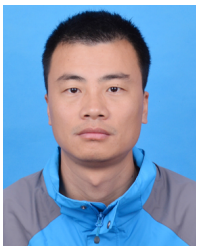
[32] S. Nandy, A. Mitra, and T. Mukherjee, "Study of PSO and Firefly algorithm based Feed-forward neural network training algorithms," in *Proc. 7th Int. Conf. Signal Process. Integr. Netw.*, Feb. 2020, pp. 908–913.

[33] W. Xinliang, C. Jianlin, M. Tianfang, L. Zhihuai, L. Na, F. Wei, L. Xuebin, Z. Liwei, and W. Jun, "Research on parallel topology analysis algorithm based on simulated annealing," in *Proc. IEEE 8th Joint Int. Inf. Technol. Artif. Intell. Conf.*, May 2019, pp. 1703–1706.

[34] A. Song, W.-N. Chen, and X.-M. Hu, "One-stage and dual-heuristic particle swarm optimization for virtual network embedding," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Glasgow, U.K., Jul. 2020, pp. 1–7.

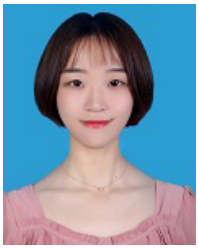
[35] X. Cheng, Z.-B. Zhang, S. Su, and F.-C. Yang, "Virtual network embedding based on particle swarm optimization," *Dianzi Xuebao (Acta Electron. Sinica)*, vol. 39, no. 10, pp. 2240–2244, 2011.

[36] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee, "How to model an Internetwork," in *Proc. 15th Annu. Joint Conf. IEEE Comput. Soc. Netw. Next Gener.*, vol. 2, Mar. 1996, pp. 594–602.



PEIYING ZHANG received the bachelor's and master's degrees from the College of Computer and Science and Technology, China University of Petroleum (East China), in 2003 and 2006, respectively, and the Ph.D. degree from the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, in 2019. He is currently an Associate Professor with the College of Computer Science and Technology, China University of Petroleum (East China). He has published multiple IEEE/ACM Transactions/Journal/Magazine articles since 2016, such as the IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING (TNSE), IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT (TNSM), IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY (TVT), IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING (TETC), *ACM Transactions on Asian and Low-Resource Language Information Processing* (TALLIP), IEEE INTERNET OF THINGS JOURNAL (IOTJ), *Computer Communications*, and *IEEE Communications Magazine*. His research interests include semantic computing, the future internet architecture, network virtualization, and artificial intelligence for networking.

YANRONG HONG is currently pursuing the degree with the College of Computer Science and Technology, China University of Petroleum (East China). Her research interests include network virtualization and artificial intelligence for networking.



XUE PANG is currently pursuing the degree with the College of Computer Science and Technology, China University of Petroleum (East China). Her research interests include network virtualization and artificial intelligence for networking.

CHUNXIAO JIANG (Senior Member, IEEE) received the B.S. degree (Hons.) in information engineering from Beihang University, Beijing, in 2008, and the Ph.D. degree (Hons.) in electronic engineering from Tsinghua University, Beijing, in 2013. He is currently an Associate Professor with the School of Information Science and Technology, Tsinghua University. His research interests include application of game theory, optimization, and statistical theories to communication, networking, and resource allocation problems, in particular space networks, and heterogeneous networks. He has served as a member for the Technical Program Committee as well as the Symposium Chair for a number of international conferences, including the IEEE ICC 2018 Symposium Co-Chair, the IWCMC 2018/2019 Symposium Chair, the WiMob 2018 Publicity Chair, the ICCC 2018 Workshop Co-Chair, and the ICC 2017 Workshop Co-Chair. He was a recipient of the Best Paper Award from the IEEE GLOBECOM, in 2013, the IEEE GlobalSIP, in 2015, the IEEE Communications Society Young Author Best Paper Award, in 2017, the IWCMC, in 2017, and the ICC, in 2019, and the IEEE ComSoc TC Best Journal Paper Award from the IEEE ComSoc TC on Green Communications and Computing, in 2018, and the IEEE ComSoc TC on Communications Systems Integration and Modeling, in 2018. He has also served as an Editor for the IEEE Internet of Things Journal, *IEEE Network*, IEEE Communications Letters, and a Guest Editor for IEEE *Communications Magazine*, IEEE Transactions on Network Science and Engineering, and IEEE Transactions on Cognitive Communications and Networking.