DSCD: Delay Sensitive Cross-Domain Virtual Network Embedding Algorithm

Peiying Zhang, Xue Pang, Yanxian Bi, Haipeng Yao, Huijiang Pan, *Member, IEEE*, and Neeraj Kumar, *Senior Member, IEEE*

***Abstract*—In the coming 5G era, more and more applications are time-delay sensitive, and the importance of low delay in fields such as telemedicine, autonomous driving and military order transmission is self-evident. However, for different application scenarios, their delay requirements are different. How to meet different Quality of service (QoS) requirements simultaneously becomes a challenge. The emergence of network virtualization (NV) technology can overcome this problem. However, it is difficult for the existing embedding algorithms to meet the requirements of virtual networks on time delay performance. Therefore, in order to study the delay problem of virtual network, we proposed a Delay Sensitive Cross-Domain Virtual Network Embedding Algorithm (DSCD-VNE) in this paper. First of all, different from the traditional virtual network embedding (VNE) algorithm, this algorithm divides the mapping process into a "node-link-node" three-stage embedding algorithm. Second, node resource metrics and set constraints are introduced in the generation phase of candidate link matrix. Third, the algorithm takes candidate link matrix as the core, and uses Kruskal minimum spanning tree to choose the link with the shortest delay for embedding. Fourth, path splitting mechanism and K-shortest path algorithm are introduced with the aim of increasing the utilization of the substrate links in link mapping stage. The simulation results show that the proposed DSCD-VNE algorithm greatly improves the performance of the algorithm compared with the traditional node-link two-stage algorithm, especially in the aspect of delay, the improvement rate is as high as 77%.**

***Index Terms*—Delay sensitive, node resource metric, path splitting mechanism, virtual network embedding, 5G network slicing.**

Manuscript received April 10, 2020; revised June 8, 2020; accepted June 22, 2020. Date of publication July 10, 2020; date of current version December 30, 2020. This work was partially supported by the Major Scientific and Technological Projects of CNPC under Grant ZD2019-183-006, partially supported by "the Fundamental Research Funds for the Central Universities" of the China University of Petroleum (East China) under Grant 20CX05017A, 18CX02139, and partially supported by the Director Foundation Project of National Engineering Laboratory for Public Safety Risk Perception and Control by Big Data (PSRPC). Recommended for acceptance by Dr. Huimin Lu *(Corresponding author: Neeraj Kumar.)*

Peiying Zhang and Xue Pang are with the College of Computer Science and Technology, China University of Petroleum (East China), Qingdao 266580, China (e-mail: 25640521@qq.com; 1103746978@qq.com).

Yanxian Bi is with the National Engineering Laboratory for Risk Perception and Prevention, Beijing 100041, China (e-mail: byxpaper@126.com).

Haipeng Yao and Huijiang Pan are with the State Key Laboratory of Networking and Switching Technology, the Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: yaohaipeng@bupt.edu.cn; panhuijiang@bupt.edu.cn).

Neeraj Kumar is with the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala 147001, India and Department of Computer Science and Information Engineering, Asia University, Taiwan and King Abdul Aziz University, Jeddah, Saudi Arabia. (e-mail: neeraj.kumar@thapar.edu).

Digital Object Identifier 10.1109/TNSE.2020.3005570

I. Introduction

In the coming 5G era, 5G will face the challenge of differentiated performance indicators in diverse application scenarios. Starting from the main application scenarios, business requirements and challenges of the mobile Internet and the Internet of things (IoT), IMT-2020 (5G) puts four 5G main application scenarios: continuous wide area coverage scenario, hot spots with high capacity scenario, low power consumption and large connection scenario, and low delay and high reliability scenario, as shown in Fig. 1. The performance challenges faced by different scenarios are different, experience rate of users, traffic density, delay, energy efficiency and connection number may all be challenging indicators for different scenarios. In order to solve the extreme differential performance requirements of diverse scenarios, it is necessary to reasonably select key technologies to meet these requirements. In some schemes, technicians combine network and blockchain technology which has rich application scenarios, and can achieve cooperative trust and consistent action among multiple agents. SDN is also an important technology to promote the development of network, AI-based SDNs combines artificial intelligence and SDN, which makes the network more human and intelligent. As one of the main sources of 5G technology innovation, network virtualization (NV) technology can also well overcome this problem.

The purpose of NV is to cut the substrate network into multiple virtual networks according to the different requirements of different scenarios [1], [2], each virtual network is oriented to different application scenarios, and the virtual networks are logically independent and do not affect each other. VNE is a research hotspot and difficulty in the field of NV. This algorithm needs to solve the problem of reasonable and effective resource allocation for different VNs under the condition of limited physical resources. Therefore, VN makes the call and allocation of network resources more flexible. Since the concept of VNE appeared, researchers have put forward a large number of VNE algorithms, but most of them are limited to receiving rate, revenue, utilization rate and other indicators when considering the mapping target, and ignore the problems that may be encountered in practical applications. In the practical application, the customer's demand for the application is no longer only satisfied with the functional requirements, but begins to pay more and more attention to the QoS of the application. Network delay is closely related to the QoS. Low network delay can bring good user experience, while different applications have different requirements for delay.

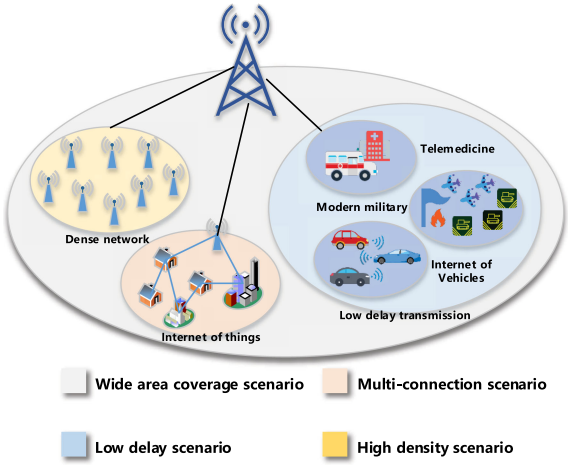


Fig. 1. The main scenes of 5G.

The high synchronization brought by the application of low delay transmission in remote surgery can help remote physicians accurately grasp the patient status and make timely guidance. Ultra-low delay network technology has been widely applied in autonomous driving and Internet of vehicles, reducing the second-level delay of sensors, which greatly reduces the braking distance and ensures the safety of drivers and pedestrians. The military networks and command information have exceedingly demand for the network performance, especially in low latency and high reliability, in the rapidly changing battlefield, no matter how advanced the combat equipment is, it will not be able to win if it cannot firmly grasp the initiative of information and communication [3]–[5].

The current VNE algorithm does not take the network delay as the main mapping target, just assumes that the delay of each physical link is the same, and evaluates the performance of the algorithm through the average path length of the mapping result. The mapping process also does not consider the delay requirements of the virtual network, but in the actual situation, due to the influence of material, length and other aspects, each physical link The delay of VNE is not the same, and different applications have different delay requirements, which requires the improvement of the current VNE algorithm to meet the needs of different applications in the actual mapping scenario. During the development of VN embedding problem, researchers have encountered three problems [6].

(1) In order to solve the VN embedding problem, majority of the traditional node-link mapping algorithms start by mapping virtual nodes, and then selecting appropriate substrate links for link mapping. Such mapping ideas have been widely recognized and applied, but they cannot meet the requirements of both solution time and VN performance. If the link delay or link cost is considered in the virtual node mapping stage, the solution time would greatly shrink, otherwise, the performance of link cost or the time delay performance is poor.

(2) In the whole process, generating the candidate link matrix is the core of the algorithm. Updating the candidate link matrix timely and accurately can improve the stability of the algorithm, reduce some unnecessary calculations, and thus shorten the solving time. The introduction of candidate link greatly improves the algorithm performance, so it is of great significance to add candidate link matrix.

(3) In order to ensure that candidate substrate nodes could satisfy the demand of virtual node resource, we are supposed to try to reduce the load request of VN resources to pieces, because too many fragments of the resources will make the rest of the substrate network resources for the allowance, as a result, it cannot continue to host new VN requests, which will reduce the acceptance rate of VN requests, thereby reducing the average long-term earnings of operators. Consequently, we need to reference some related concept to solve these problems.

In order to solve the above problems, this paper proposes a delay sensitive cross domain virtual network embedding algorithm. The main contribution of this paper can be concluded as follows.

(1) The proposed method divides the embedding process into "node-link-node" three steps, which is different from the traditional "node-link" two stages embedding algorithms. The use of three-stage algorithm focuses on the link mapping which has more influence on the delay in the virtual network.

(2) In the stage of candidate link matrix generation, we introduce the node resource metrics and constraints with the purpose of improving the quality of node selection. At the same time, the advantage of candidate link matrix is fully utilized.

(3) In the link mapping stage, the path segmentation mechanism and k-shortest path algorithm are used for link mapping. The simulation demonstrates that the proposed method can reduce the fragmentation of substrate resources, and increase the VN request acceptance ratio.

The remainder of this paper is organized as follows. Section II retrospects the existing algorithms for VNE. Section III recommends the network model and problem statement. In Section IV, we introduce the steps of the proposed algorithm. In Section V, we describe the detail of the DSCD-VNE method that we proposed. The performance of our method and other methods is evaluated in Section VI. Section VII concludes this paper.

II. Related Works

In this section, we will review these existing literatures [7]–[13] on multi-domain virtual network embedding (MVNE) algorithms. Prior studies [14]–[17] on virtual network embedding have mostly addressed the single domain virtual network embedding problems. Many existing researches have combined network and artificial intelligence to realize the intellectualization of network. The authors of literature [18] combine the Internet of vehicles with artificial intelligence, which makes the driverless technology more intelligent. In addition, emotional communication technology is applied to social robots to make services more personalized. Based on the cloud/fog-computing mode and the Internet of things artificial intelligence service framework, the authors of literature [19] propose a cross domain vehicle driving solution, which can effectively improve the traffic performance and make the

automatic driving technology more mature. Literature [20] uses a new classification method to classify the research in recent years, and summarizes these algorithms. From this paper, we learn a lot of new ideas and gain a lot of inspiration. These existing algorithms can be roughly partitioned into two categories including the distributed algorithms and the centralized algorithms.

A. The Distributed MVNE Algorithms

The distributed agents are used in [21] to cooperatively accomplish the VNE process according to the substrate resource information and the shortest path parameters. The authors of [22] propose a privacy-preserving multi-domain attack detection scheme for SDNs called Predis. This scheme adopts a simple and efficient algorithm as the detection algorithm, which can effectively protect privacy and improve the security performance of the network. The author of [23] first establishes the calculation model of task completion delay in Mobile edge computing (MEC), at the same time, the energy consumption of different equipment in MEC is analyzed. This paper proves that this problem belongs to NP-hard problem and then presents a solution algorithm based on bend decomposition. A framework based on software defined network (SDN) is introduced to deploy the algorithm. The simulation results indicate that this algorithm can improve the performance both in energy consumption and acceptance ratio.

Machine learning is very popular in the field of smart cities [24], which can collect large amounts of data from various IoT devices. The authors of [17] present a novel three dimension resource constraints based virtual network embedding algorithm. With the application of machine learning method in computer networks [25], [26], some researchers investigated the reinforcement learning based on VNE algorithms [27], [28].In modern intelligent transportation system, dynamic pricing plays an important role in solving congestion control, peak load reduction and mobility management of traditional or electric vehicles. By optimizing vehicle path planning, the delay of congestion can be reduced and the construction of social ecological environment can be promoted. Based on these facts, the authors of [29] present a systematic review of dynamic pricing techniques for Intelligent Transportation System (ITS) in smart cities. This paper discusses the problems solved by dynamic pricing technology, the importance of various evaluation parameters, the limitations of dynamic pricing technology and its application.

The authors of [30] propose a platform architecture for virtual network (VN) provisioning across multiple domains, this architecture decomposes VN provisioning into several stages in order to solve the impact of limited exposure on resource discovery and resource allocation. (1) resource discovery (2) resource allocation and (3) VN instantiation. With the rapid development of mobile devices and cloud computing, users can enjoy various cloud computing services on mobile devices. However, the openness of wireless communication makes security and privacy protection difficult. To solve this problem, the authors of [31] propose an efficient privacy-aware authentication (PAA) scheme for mobile cloud computing services. The security analysis shows that the proposed PAA scheme has less computation and communication overhead and can effectively reduce the delay.

The authors of [32] devise a distributed protocol witch guarantee the service providers (SPs) competitive prices. First, the authors send the VN request according to its location constraints. Second, the autonomous domain would pre-embedding the VN request and return the embedding price for its embedding solution. Third, all domains coordinate the whole virtual network embedding operation using distributed bidding manner. The authors of [33] investigated the tricky problem of multi-domain VN embedding with limited information disclosure (LID). The major contribution of their work is to propose a traffic matrix based virtual network embedding architecture that facilitates VN request partitioning process under LID. The authors of [34] propose a Markov Reward based Resource-Latency Aware Heuristic for the VNE problem, and their solutions around the new concept of similarity between virtual nodes and substrate nodes, using the theory of markov reward, for each of the physical and virtual nodes, defined a set of metrics for capturing the node number of resources in the neighborhood and the close degree between nodes, and achieved good effect in performance. The authors in [35] introduce an inter-domain virtual network (VN) embedding policy for revenue maximization. More precisely, each substrate infrastructure (InP) model other domain nodes as a domain node with an aim to construct an abstracted network topology. Then, each InP rely on genetic algorithm to solve the embedding problem with the minimal cost. Third, the virtual network provider (VNP) choose the InP with the minimal embedding cost to partition the VN request according to its embedding solution. Finally, the respective InP accomplish the VN embedding relying on the specific single domain VN embedding algorithm.

The existing cross domain virtual network embedding algorithm has been considered in many aspects, such as energy consumption, security, bandwidth and so on, but few papers consider the problem of delay. In the intelligent city, Internet of things and other Internet systems, the importance of time delay can not be ignored. Based on this, this paper fully considers the related problems of time delay and realizes the time delay sensitivity algorithm, which is a new idea.

B. The Centralized MVNE Algorithms

Virtual networks typically span multiple domains and can be controlled by different infrastructure providers (InPs). However, in order to achieve efficient VN mapping across multiple domains, a global view is required. The authors in [36] propose a minimal cost based multi-domain virtual network embedding algorithm. First, the candidate substrate node set is constructed according to the constraints of virtual network request. Second, a feasible substrate path set is computed for each pair of nodes corresponding to each virtual link. Third, on the principle of Kruskal algorithm, the substrate path with the minimum weight is chosen meanwhile the

corresponding virtual nodes are embedding in a harmonious manner. The mapping of VNE in a substrate domain has been well studied, but the demand for virtual networks is growing rapidly, requiring extensive virtual networks across multiple domains. At present, the researches of this problem primarily concentrate on dividing a VN into sub-VN of each domain, and little attention has been paid to the associative relationships between intra-domain links and inter-domain links.

SDN is a new framework to realize network virtualization, which can effectively improve the efficiency of virtual network embedding. Industrial Internet of things (IIoT) is widely used in transportation, health-care, manufacturing and energy management for smart grids, and in these feilds, SDN can well replace the traditional network infrastructure based on TCP/IP. The authors of [37] present a SDN-enabled multi-attribute-based secure communication model for smart grid in IIoT environment, which is more appropriate than most secure communication solutions to address the problems in those areas. In order to optimize the utilization of overall resources, the authors of [38] propose a multi-domain link mapping method combining intra and peer link mapping. This proposed algorithm decomposes the VN problem into two steps. In the first step, VNP decomposes nodes and optimizes node embedding. In the second step, VNP performs some simplified VNE subsolutions, each of which optimizes domain-dependent intra and peer link mappings. This approach is based on the current Internet architecture and therefore easy to deploy. The authors of [39] propose a survivable virtual network mapping algorithm (IntD-GRC-SVNE) that implements multi-domain mapping in NV, the main contribution of this algorithm is to map the virtual communication network to different domain networks. Therefore, it provides backup resources for the virtual link and improves the survivability of the special network. Moreover, this algorithm also introduces global resource capacity (GRC), which makes the network load become more balanced and greatly improves the VN request acceptance rate.

The authors in [40] introduce a two-stage multi-domain VN embedding policy named DPSO-K, which uses the resource bidding method to implement the VN request partition relying on the node resources and inter-domain link bandwidth resources. Finally, the authors use Kruskal minimum spanning tree to accomplish the intra-domain VN embedding process. This work adopted the manner of centralized management and distributed control with an aim to efficiently use the substrate resources. The authors of [41] propose some interesting resource allocation problems in network virtualization and also propose a resource allocation framework(RAiNV) to deal with these problems. First, this article describes a single VNP scenario, and then extends to multiple VNPs scenarios. The main idea of RAiNV is to utilize more information in VNP. And use them for resource allocation. In RAiNV, the focus is on the four pieces of information available in VNP. This information includes pricing, node location, node capacity, and link capacity between InPs. Pricing information can be static or dynamic. Areas such as intelligent transportation systems, intelligent electronic health care, and intelligent education have high time-delay requirements. In these applications,

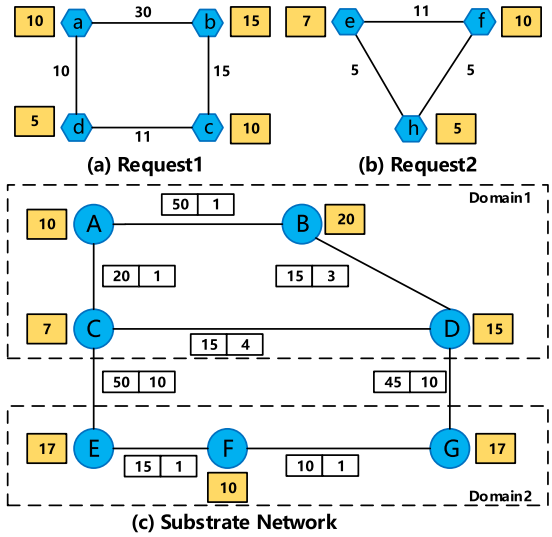


Fig. 2. The diagram of substrate network and a virtual network request.

various devices can be identified using radio frequency identification (RFID) tags, which can be used to transmit the collected data to other local or remote objects in different geographical locations. However, in the existing RFID-based authentication system, the adversary may access the information of the RFID tag, which not only violates privacy, but also may cause forgery problems. In order to solve the above problems, the authors of [42] propose a lightweight and privacy-preserving RFID authentication scheme for distributed IoT infrastructure with secure localization services for smart city environment. Compared with the existing scheme, it can better optimize the execution time and reduce the delay.

Many of the existing algorithms are centralized. In these models, there is a centralized manager who is responsible for centralized and published information. The algorithm proposed in this paper does not use the centralized mapping model, but divides the mapping process into three steps, each step is independent, but we will consider adding an agent as middlemen in our future work to solve this kind of problem.

III. Network Model and Problem Statement

A. Virtual Network Request Model

Each virtual network request (VNR) can be modelled as an undirected weighted graph, which is composed of virtual nodes and virtual links.

As Fig. 2 depicted, the virtual network request (a) contains 4 virtual nodes {a, b, c, d}, each virtual node has a corresponding CPU resources demand (the number in the rectangle). The virtual network request (a) contains four virtual links {a-b, a-d, b-c, c-d}, each link has a corresponding bandwidth resources demand (the number next to the link).

The undirected weighted graph that represents the VNR is shown below:

$$G_v = \{N_v, L_v\}, \tag{1}$$

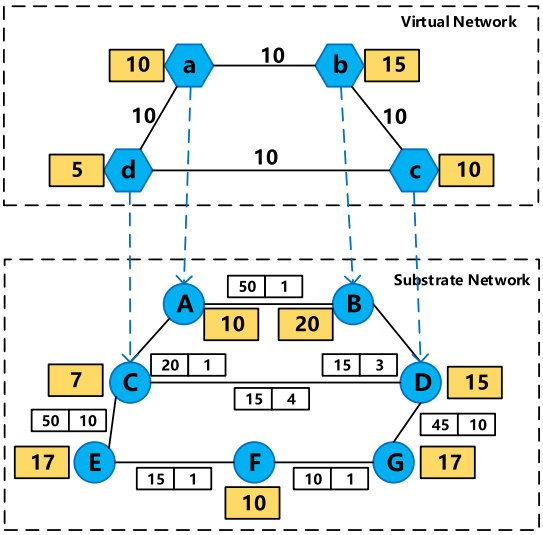


Fig. 3. The diagram of virtual network mapping process.

where N_v=$\{n_V^1, n_V^2, n_V^3...\}$ represents the set of nodes, the CPU resources demand of virtual node $n_V^i \in n_V$ can be expressed as $CPU(n_V^i)$. And the formula L_V represents the set of virtual links, the bandwidth resources demand of virtual link $(n_V^i, n_V^j) \in L_V, i \neq j$ can be expressed as $Bw(n_V^i, n_V^j)$.

B. Substrate Network Model

The substrate network model and resource condition are shown in Fig. 2. The substrate network (c) contains 7 substrate nodes {A, B, C, D, E, F, G}, each substrate node has a corresponding available CPU capacity (the number in the rectangular box). The substrate network (c) also contains 8 substrate links {a-b, a-c, b-d, c-d, c-e, d-g, e-f, f-g}, each substrate link has a corresponding available bandwidth resources (the number next to the link) and a link delay (the number in parentheses next to the link bandwidth resources).

Similarly, the substrate network can also be modelled as an undirected weighted graph:

$$G_S = \{N_S, L_S\}, \tag{2}$$

where $N_S = \{n_S^1, n_S^2, n_S^3, n_S^4, n_S^5...\}$ represents a collection of nodes in a substrate network, the available CPU resources of the substrate node $n_S^i \in N_S$ can be expressed as $CPU(n_S^i)$. And L_S represents the set of substrate links, the available bandwidth resources of substrate link $(n_S^i, n_S^j) \in L_S, i \neq j$ can be expressed as $Bw(n_S^i, n_S^j)$, the delay of the link can be expressed as $Delay(n_S^i, n_S^j)$.

C. Virtual Network Embedding Model

The VNE problem can be expressed as a mapping process, $VNE : G_S\{N_S, L_S\} \rightarrow G_V\{N_V, L_V\}$, Fig. 3 shows the embedding result of virtual network request 1. The mapping process is often divided into two mapping stage:

(1) Node Mapping Stage: In the node mapping stage, virtual nodes are allocated to heterogeneous substrate nodes and resource constraints on nodes are satisfied. For the same virtual network request, if virtual node n_V^i is mapped to substrate node n_S^i, it shall satisfy:

$$CPU\left(n_S^i\right) \geq CPU\left(N_V^j\right), \tag{3}$$

$$n_S^i \in VnMatch\left(n_V^i\right). \tag{4}$$

Formula (3) represents that the available CPU resources of substrate node n_S^j mapped by virtual node n_V^i must be bigger than or equal to the CPU resources demand of virtual node n_S^j, that is, substrate node n_V^i should meet the resources demand of virtual node n_V^i. Formula (4) represents that n_S^j is in the list of candidate mapping nodes of n_V^i, that is, n_S^j must satisfy specific constraints that $VnMatch(n_V^i)$ generation follows a particular algorithm.

In addition, for the same virtual network request, each substrate node of the substrate network can only be mapped once at most, that is, virtual node and substrate node should correspond one to one.

(2) Link Mapping Stage: In the link mapping stage, virtual links are assigned to a loopless underlining path on the link and resource constraints on the link are satisfied. For the same virtual network request, if virtual link (n_V^u, n_V^v) is mapped to substrate link (n_S^m, n_S^n), it shall satisfy:

$$\sum Bw\left(n_S^m, n_S^n\right) \geq \sum Bw\left(n_V^u, n_V^v\right), \tag{5}$$

$$Bw\left(n_S^m, n_S^n\right) \geq Bw\left(n_V^u, n_V^v\right). \tag{6}$$

Formula (5) indicates that when mapping, the total bandwidth resources of the substrate network should be bigger than or equal to the total bandwidth resources requested by the virtual network, formula (6) represents that the resource constraints of link mapping, that is, the bandwidth resources of mapped substrate link (n_S^m, n_S^n) must satisfy the bandwidth resources demand of virtual link (n_V^u, n_V^v).

D. Optimization Objectives

The performance metric to evaluate the compared algorithms is delay metric of the virtual networks, which can be obtained by the product of bandwidth resources and link delay. The $Delay(G_v)$ for evaluating the delay performance of the virtual network G_v can be represented as follows:

$$Delay(G_v) = \sum Bw(n_V^u, n_V^v) * Delay(n_S^m, n_s^n), \tag{7}$$

where $Delay(G_v)$ represents the delay performance of the virtual network G_v, $Bw(n_V^u, n_V^v)$ represents the bandwidth resources demand of the virtual link (n_V^u, n_V^v), $Delay(n_S^m, n_s^n)$ represents the delay of substrate link (n_S^m, n_s^n), we define the delay of a virtual network by the sum of the product of bandwidth and delay, and the smaller $Delay(G_v)$ value indicates the better delay performance.

Therefore, our objective is to find a optimal VNE strategy with the aim of minimizing the average $Delay(G_v)$ for the virtual network G_v.

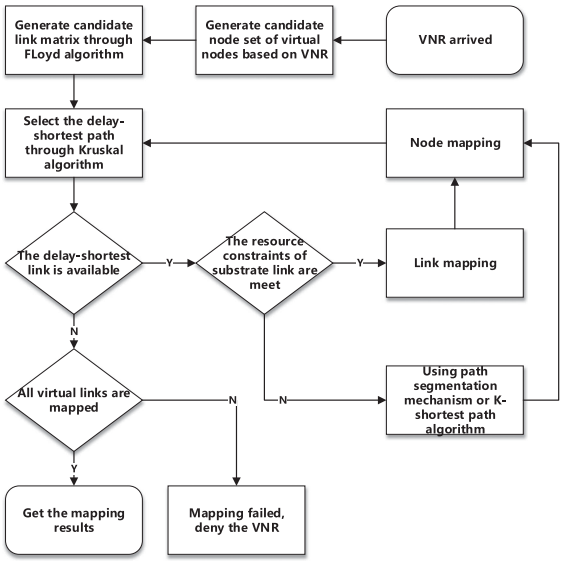


Fig. 4. The flow chart of DSCD-VNE algorithm.

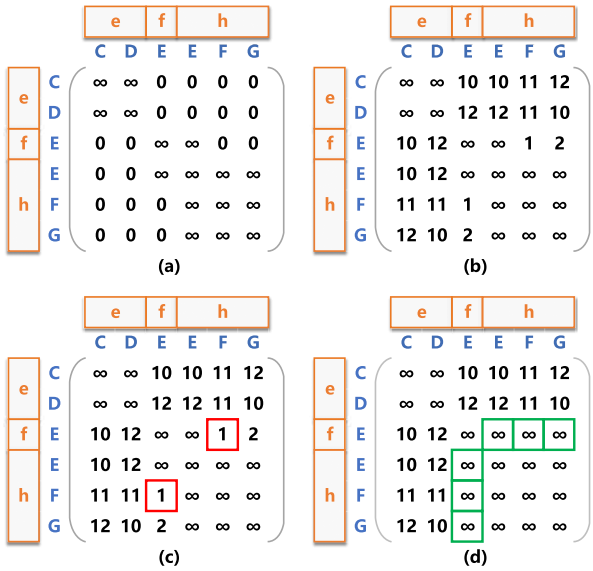


Fig. 5. An example of virtual network mapping process.

IV. The Embedding Steps of DSCD-VNE Algorithm

In this section, we will introduce the mapping steps of DSCD-VNE algorithm. Figure 4 shows the flow chart of the algorithm.

Step 1: Candidate substrate nodes selection stage Each virtual node needs to have a candidate node set $VnMatch({n_V}^i)$, and each $VnMatch({n_V}^i)$ should at least meet the formula (3) and formula (4), that is, the candidate physical nodes should at least meet the CPU resources demand of the corresponding virtual node. In addition, at most a certain number of candidate nodes are selected to enter the set, so as to avoid the too long set length seriously affecting the solution speed.

If the candidate node set of a virtual node is empty, that is, there is no matching physical nodes as the candidate physical nodes of the virtual node, exit the algorithm and refuse the virtual network request.

After the selection of candidate nodes, the first step of the algorithm is completed, and then the link mapping stage is entered.

Step 2: Link mapping stage

In the link mapping stage, we introduce the candidate link matrix. Firstly, the shortest path $(n_S^m, n_S^n) = (n_S^m, n_S^a, n_S^b, n_S^n)$ is calculated by the Kruskal algorithm, and the virtual link (n_V^u, n_V^v) is tried to map to the physical link (n_S^m, n_S^n). According to the shortest physical link selected, view the sub path (n_S^a, n_S^b) with the smallest resources in the link. The bandwidth resources of the sub path is the minimum resources of the selected candidate physical link, and the maximum virtual link resources demand that the whole physical link can carry, as shown in formula (8):

$$Bw(n_V^u, n_V^v) \leq min\{Bw(n_S^m, n_S^n), Bw(n_S^a, n_S^b), Bw(n_S^b, n_S^n)\}, \tag{8}$$

If the above formula is true, the third stage will be entered, otherwise, the link mapping will be carried out through path segmentation or k-shortest path algorithm. These two algorithms are detailed in Section 5.

Step 3: Node mapping stage

After the link mapping is completed, the nodes at both ends of the link are mapped, and the mapping process is completed.

An example: We will introduce the mapping steps of DSCD-VNE algorithm with a concrete example. Take the VNE model in Fig. 2 as an example, and assume that the virtual network request 2 arrives. First, we should determine the candidate set $VnMatch({n_V}^i)$, suppose that $VnMatch(e) = \{C, D\}, VnMatch(f) = \{E\}, VnMatch(h) = \{E, F, G\}$.

According to the candidate physical node set of virtual nodes e, f, h, the candidate link matrix can be initialized, as shown in Fig. 5(a). For the same virtual node, the corresponding link elements in the candidate link matrix are all counted as infinity, indicating that they are not reachable; meanwhile, the corresponding link elements of the same physical node are all counted as infinity, and other elements are initialized as 0.

After the initialization of candidate link matrix, based on the idea of Kruskal minimum spanning tree, traverse the candidate link matrix and select the link with the shortest path as the link to be mapped, as shown in Fig. 5(b) and Fig. 5(c). After link selection, we should judge the node and link status: (1) Judge the mapping state of nodes: virtual nodes f, h and physical nodes E, F are all not mapped, so they can be mapped. (2) Judge the link resource status: the bandwidth resources demand of virtual link (f, h) is 5, while the available bandwidth resources of physical link (E, F) is 15, which meets the formula (6), therefore, the virtual link (f, h) can be mapped to physical link (E, F). After the mapping, the available bandwidth resources of the physical link are modified according to the bandwidth resources demand of the virtual link in the virtual network request: $Bw(E, F) = Bw(E, F) - Bw(f, h)$.

As a virtual link mapping is completed, the candidate link matrix needs to be updated, as shown in Fig. 5(d). For the

mapped virtual link, the other candidate physical links should be set to infinity, which will not be considered in the next algorithm calculation of the shortest path.

Virtual node f is mapped to physical node E, virtual node h is mapped to physical node F, and then the mapping state of the two nodes is modified.

Enter the next cycle until the end of the algorithm.

At the end of the algorithm, the condition is that the number of virtual links mapped is equal to the number of virtual links requested by the virtual network. When the algorithm exits, if the condition is satisfied, the mapping is successful, if the condition is not satisfied, the virtual network request will be rejected.

V. Delay Sensitive Cross-Domain Virtual Network Embedding Algorithm

The DSCD-VNE (Delay Sensitive Cross-Domain Virtual Network Embedding) algorithm expands the traditional two-stage process of "link first, node later" into a three-stage process of "node-link-node". In the following sections, we describe the details of several key algorithms.

A. Candidate Substrate Node Selection Algorithm

The selection of candidate substrate node set directly determines the generation of candidate link matrix, which has a big impact on the performance of the DSCD-VNE algorithm. Therefore, it is of great significance to choose a suitable solution algorithm for the selection of the set of candidate substrate nodes.

In order to make the choice of candidate substrate nodes could satisfy the requirement of virtual node resources, we should reduce the load request of virtual network resources in pieces, because too many fragments of the resources will make the rest of the substrate network resources for the allowance, therefore, it cannot continue to host new virtual network requests, which will reduce the acceptance rate of VN requests, and thus reduce the average long-term earnings of operators. In view of the above problems, the algorithm uses the concept of node resource measurement (NRM) for reference, and uses the constraint mechanism of the candidate node set to avoid the reduction of acceptance rate, which caused by repeated selection of nodes.

Our work is similar to one in [34], in order to better measure node resources, we redefine a node resource measure (NRM) as follows.

$$NRM(n_V^i) = \alpha * \sum_{j \neq i} Bw(n_V^i, n_V^j) + \beta * CPU(n_V^i), \quad (9)$$

$$NRM(n_S^i) = \alpha * \sum_{j \neq i} Bw(n_S^i, n_S^j) + \beta * CPU(n_S^i), \quad (10)$$

$$\alpha + \beta = 1. \quad (11)$$

The formula $NRM(n_V^i)$ is the node resource measurement of virtual node n_V^i, and $\sum_{j \neq i} Bw(n_V^i, n_V^j)$ is the sum of bandwidth resources of all virtual links that connected by virtual node n_V^i, which is the degree of n_V^i; and $NRM(n_S^i)$ is the node resource measurement of substrate node n_S^i, $\sum_{j \neq i} Bw(n_S^i, n_S^j)$ is the sum of bandwidth resources of all virtual links connected by virtual node n_S^i, which is the degree of n_S^i. Parameter α is the link resource weight of equation, and parameter β is the node resource weight. The relationship between α and β is shown in equation (11). According to the definition of equation (9)(10), the closer the NRM is, the more similar the virtual node is to the substrate node in the network environment. Taking such a node pair as a candidate node pair is of great help to increase the acceptance rate of the virtual network and reduce the resource fragmentation generated by the mapping.

Algorithm 1: Candidate Substrate Node Selection Algorithm

Input: Virtual Nodes:vnode,Substrate Nodes Resource Metric List;
Output: Virtual Node Candidate Set: VnMatch;

```
if VnMatch(vnode).size() < MaxMatch then
    VnMatch(vnode).push_back(snode);
else if SnMatch(snode) < MaxMatched then
    ucnode = VnMatch(vnode).unclosestPos();
    if NRM(snode) close to NRM(vnode) then
        VnMatch(vnode).unclosestPOS() = snode;
        SnMatch(snode) + +;
        SnMatch(ucnode) − −;
    end if
end if
Return VnMatch(vnode);
```

In the selection stage of candidate substrate nodes. Firstly, we should pay attention to the length of substrate candidate node set, on the one hand, if the number of candidate substrate nodes of each virtual node is small, then the available links in the candidate link matrix also meet the requirements of time delay characteristic optimization. On the other hand, if the number of candidate substrate nodes of each virtual node is large, the order of candidate link matrix to be processed during the execution of the algorithm will also be large, then the solving time of the algorithm may not be able to meet the requirements of operators who want the service to go online quickly when providing the service. Secondly, we should pay attention to the number of each substrate node which has been selected, if a substrate node is selected by multiple virtual nodes as the candidate node at the same time, when the node is mapped by a virtual node, many other virtual nodes' selection scope is narrow, the choice of the whole algorithm was significantly narrowed down, and may even lead to logically reject a virtual network request.

Therefore, we designed the two parameters MaxMatch and MaxMatched. MaxMatch limits the amount of substrate nodes that can be the candidate nodes for each virtual node, while MaxMatch limits the number of times that each substrate node can be matched by a set of candidate nodes. The choices for MaxMatch and MaxMatched can be constant, but it is recommended to adjust for the size of the network.

The Algorithm 1 describes the detailed steps of selecting candidate substrate node set.

B. Link Mapping Algorithm Using Path Splitting Mechanism

The link mapping stage of path splitting or the link mapping stage of k-shortest path algorithm are both designed to improve the acceptance rate of virtual network requests, thus improve operators' revenue and increasing the stage long-term average ratio of income and expenditure.

Because the substrate link bandwidth resources is insufficient, some links with "low delay and few resources" cannot be fully utilized, so cannot achieve the delay optimization. In order to solve this problem effectively, this paper introduces the link mapping mechanism of path splitting.

For most VNE algorithms, virtual link requests are mapped to a substrate link, which requires that this substrate link needs sufficient resources to carry the resource requests of a whole virtual link, but the link with low delay does not mean the bandwidth resources are sufficient. Therefore, in order to make full use of low-delay links, in the link mapping stage, it is no longer limited that one virtual link can map to only one substrate link, but allows the same virtual link to map to multiple substrate links at the same time.

At the time of candidate link stage to choose the shortest link, the stage want to map a virtual link (n_V^u, n_V^v) to the substrate link $(n_S^m, n_S^n) = (n_S^m, n_S^a, n_S^b, n_S^n)$ which with the shortest time delay, but it does not meet the demands of bandwidth, at this point, we believe that there are still other substrate links connecting m and n between the virtual node pair (u, v) and the corresponding substrate node pair (m, n), so we should not refuse the virtual network request.

For the above problems, the algorithm 2 proposed in this paper is mandatory to complete the link mapping and subsequent node mapping processes. By traversing substrate link $(n_S^m, n_S^n) = (n_S^m, n_S^a, n_S^b, n_S^n)$, the minimum bandwidth resource minRes on the current path can be obtained according to equation (12).

$$minRes = min\{Bw(n_S^m, n_S^a), Bw(n_S^a, n_S^b), Bw(n_S^b, n_S^n)\}, \tag{12}$$

$$Bw\left(n_S^i, n_S^j\right) = minRes, \tag{13}$$

$$Bw(n_V^u, n_V^v) = Bw(n_V^u, n_V^v) - minRes. \tag{14}$$

This mapping will first allocate the minRes for the bandwidth resource requests of (n_V^u, n_V^v), that is, the virtual link (n_V^u, n_V^v) is mapped to the substrate link $(n_S^m, n_S^n) = (n_S^m, n_S^a, n_S^b, n_S^n)$ in consideration of completing the mapping of the virtual link for several times. This mapping does not satisfy bandwidth resource demands, so there will be at least one subpath $(n_S^i, n_S^j) \in (n_S^m, n_S^a, n_S^b, n_S^n)$ as shown in equation (13), which will be regarded as a broken path due to resource exhaustion.

After link mapping, the node mapping is performed, as $u \rightarrow m, v \rightarrow n (or u \rightarrow n, v \rightarrow m)$. After node mapping, the virtual network requests need to be modified, as shown in equation (14), and the shortest delay path of the current substrate node pair needs to be re-calculated by Floyd algorithm to enter the next iteration.

The path splitting Algorithm is shown in Algorithm 2.

Algorithm 2: Link Mapping Algorithm using Path Splitting Mechanism

Input: Virtual Link: ${L_V}^{uv}$, Substrate Link: ${L_S}^{mn}$, Virtual Link Request metric: Res

Outout: Actually Used Bandwidth Resource: $UsedRes$

```
minPath = GetFloydResult(L_S^mn);
minRes = GetminRes(minPath);
if Res < minRes then
    MapLinks(L_V^uv, L_S^mn, minRes);
    UsedRes = Res;
else
    MapLinks(L_V^uv, L_S^mn, minRes);
    UsedRes = minRes;
end if
return UsedRes;
```

C. Link Mapping Algorithm using K-shortest Path Algorithm

Although the adoption of path segmentation algorithm can achieve a relatively high acceptance rate of virtual network requests, not all the substrate network devices support such an algorithm, especially in cross domain mapping scenarios, the protocols supported by InPs in different domains are not the same, and it is difficult to unify such special cases as path segmentation. Therefore, K-shortest path algorithm is introduced as a backup link mapping algorithm.

The K-shortest path algorithm is used by most mainstream VNE algorithms in link mapping stage. The basic idea is to calculate up to K shortest paths for the node pair after the mapping node pair is determined, and then map from small to large in order. If the resources of the substrate link meets the resource requirements of the virtual link, the mapping is completed. If none of the K links can meet the resource requirements of the virtual link, the virtual network embedding fails and the virtual network request is rejected. For the value of K, the larger K is, the higher the acceptance rate of virtual network request is and tends to be stable, but the solution speed slows down with the increase of K, therefore, we take 5 as the value of K.

The k-shortest path algorithm is shown in Algorithm 3.

D. DSCD-VNE Algorithm

Based on the Algorithm 1, Algorithm 2 and Algorithm 3, we give the mapping procedure of our algorithm called DSCD-VNE in Algorithm 4.

E. Time Complexities Analysis

We use N^V and L^V to represent the number of virtual nodes and virtual links of each VN request respectively, meanwhile, we use N^S and L^S to represent the number of substrate nodes and substrate links of substrate network respectively.

Algorithm 3: Link Mapping Algorithm using K-shortest Path Algorithm

Input: Virtual Link: ${L_V}^{uv}$, Substrate Link: ${L_S}^{mn}$, Virtual Link Request metric: Res
Output: Is Link Map a Success

```
count = 0;
for count < K do
    Path = GetminPath(L_S^{mn}, count);
    minRes = GetminRes(Path);
    if Res < minRes then
        LinkMap(L_V^{uv}, L_S^{mn}, Res);
        return true;
    else
        count++;
    end if
end for
return false;
```

Algorithm 4: The Proposed DSCD-VNE Algorithm

Input: Virtual Network Request $G_V = \{N_V, L_V\}$, Substrate Network Topology $G_S = \{N_S, L_S\}$, $iteration$
Output: VNE Embedding Policy

```
VNetwork vn;
SNetwork sn;
Iteration Number iteration;
count=0;
for all the unmapped virtual links in VNR do
    for count < iteration do
        generate a candidate set of substrate nodes using Algorithm 1;
        calculate the shortest delay path between substrate nodes using Floyd algorithm;
        generate the candidate link matrix;
        choose the shortest delay link for mapping;
        if the shortest delay link is available then
            if the substrate link resources satisfy the constraints then
                node mapping;
            end if
        else
            break;
        end if
        return mapping_policy;
    end for
    count++;
end for
```

The Time Complexity of DSCD-VNE: Based on the data, we can find that the time complexity of algorithm 1 is $O(|N^V|)$, and the time complexity of Floyd algorithm is $O(|L^S|^3)$, therefore, the time complexity of algorithm 2 is $O(|L^S|^3)$, the time complexity of algorithm 3 is $O(k|L^S|^3)$, the time complexity of DSCD-VNE is $O(|L^V||L^S|^3)$.

The Time Complexity of MC-VNE: The time complexity of MC-VNE is $O(|L^V||N^S|^2)$.

The Time Complexity of VNE-PSO: The time complexity of VNE-PSO is $O(|N^V||N^S|^2)$.

TABLE I
THE COMPARED FOUR METHODS

Notation	The idea of algorithms
DSCD-VNE	The set of candidate nodes and candidate link matrix are generated according to the node resource measurement. According to the Kruskal spanning tree algorithm, the shortest substrate path is selected for link mapping, and the path splitting mechanism is adopted. Node mapping process is carried out subsequent to the link mapping process.
VNE-PSO	Select candidate substrate nodes for virtual nodes based on specific rules, and then the particle swarm optimization algorithm was used for node mapping with the aim to find the particle with the lowest fitness value, link mapping adopted the shortest path algorithm.
MC-VNE	Select substrate nodes that meet the resource requirements as the candidate nodes, and then select the shortest substrate path based on the principle of Kruskal algorithm, the node mapping is carried out in a harmonious way.
MP-VNE	First, the candidate nodes are selected by using the estimated mapping cost. Then, the centralized hierarchical multi domain virtual network mapping architecture is used to collect the information of each domain and process it synthetically, and the particle swarm optimization algorithm is used to optimize the experimental results.

The Time Complexity of MP-VNE: The time complexity of node embedding is $O(|N^V|)$, and the time complexity of link embedding is $O(|N^S|^2)$, thus the time complexity of MP-VNE is $O(|N^V||N^S|^2)$.

VI. Simulation Experiments and Analysis

In this part, we describe the settings of the simulation environment and give the simulation results. We measure the performance of our method against other methods by three evaluation criteria, including the time delay, the virtual network request acceptance rate, and the average running time. This section will compare the proposed DSCD-VNE algorithm with the MC-VNE algorithm, the VNE-PSO algorithm and the MP-VNE algorithm. The differences on these four algorithms are listed in Table I.

A. Experimental Environment Settings

The preparation of the simulation program uses C++ as the programming language, and the development platform is JetBrains CLion. The computer used in the simulation experiment is configured with 8GB memory, 64-bit Win10 operating system, and Intel Core i5-8300h processor. We use the GT-ITM[43] to generate the topologies of SNs and VN requests. In the process of simulation experiment, the arrival of virtual network requests obeys Poisson distribution, reaching about 1000 virtual network requests in 2200 unit time, and the lifetime of each virtual network request obeys negative exponential distribution. This part is also added in the article. Table II shows the parameter settings.

B. Experimental Results and Analysis

In this section, all experiments are divided into two parts, the first part includes two experiments, they are designed to

TABLE II
THE SETTINGS OF PARAMETERS

Parameter Items	The Range
the number of substrate domains	4
the number of substrate nodes	120
the CPU resources of substrate nodes	U[100,300]
the bandwidth resources of substrate links	U[1000,3000]
the delay of substrate links	U[1,10]
the number of virtual nodes	U[2,10]
the CPU resources demand of virtual nodes	U[1,10]
the bandwidth resources demand of virtual links	U[1,10]
link resource weight parameter α	0.5
node resource weight parameter β	0.5

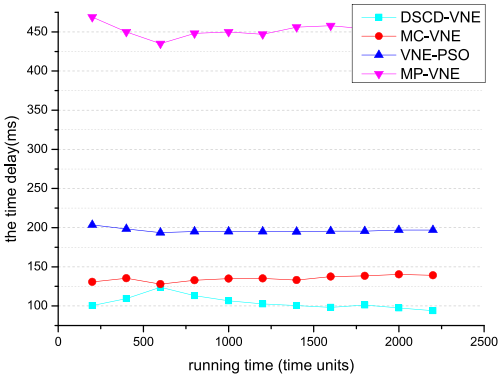


Fig. 6. Time delay performance comparison of the four algorithms.

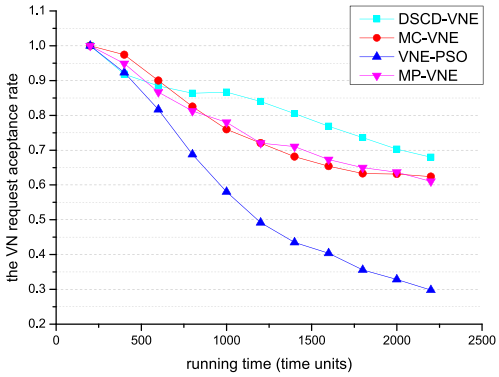


Fig. 7. Virtual network delay comparison of the four algorithms.

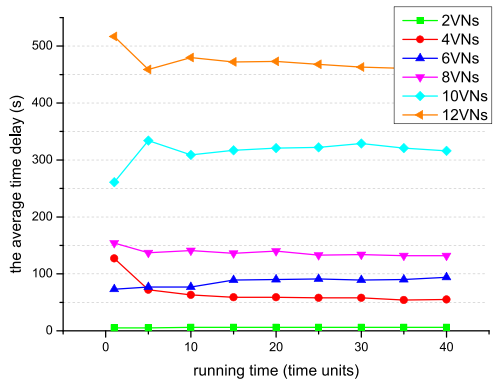


Fig. 8. The average time delay of different virtual network size.

compare the performance of the four algorithms, including time delay and virtual network request acceptance rate. The second part includes three experiments, whose purpose is to verify that the DSCD-VNE still performs well with virtual network grow in size.

In this part, we designed two experiments to compare the time delay performance and virtual network request acceptance rate of the four algorithms.

Experiment 1: The time delay performance comparisons

In order to compare the time delay performance of the four algorithms at different time points, simulation experiments are carried out in the simulation environment. The experimental results are shown in Fig. 6. According to the formula (7) for calculating the experimental optimization performance, it can be concluded from Fig. 6 that the DSCD-VNE algorithms have better time delay performance than the other three algorithms, and the delay of MP-VNE is much higher than the other three algorithms. The main reason for this result is that DSCD-VNE algorithm uses the appropriate method of selecting candidate nodes, and fully considers the delay of the link when selecting the candidate substrate links, while other algorithms consider the comprehensive performance of each attribute, and do not pay too much attention to the delay. Therefore, DSCD-VNE algorithm is far superior to the other three algorithms in the delay performance. Through quantitative analysis of specific data, our method DSCD-VNE is 22.71% lower than MC-VNE in terms of average delay, 46.86% lower than PSO-VNE and 77% lower than MP-VNE.

Experiment 2: Virtual network request acceptance rate comparison of the four algorithms

The results of experiment 2 are shown in Fig. 7. In order to compare the virtual network request acceptance rate of the four algorithms, experiment 2 compared the virtual network request acceptance rate of the three algorithms at each time points. Obviously, as shown in the figure, as time goes on, the acceptance rate of virtual network requests is getting lower and lower. Since we introduced the path splitting mechanism in DSCD-VNE, the resource fragments generated by hosting virtual network requests became more orderly, making the substrate network enable to host more virtual network requests. Therefore, the virtual network requests acceptance rate of DSCD-VNE decreased more gently with the passage of time, obviously better than the other three algorithms. Through quantitative analysis of specific data, our method DSCD-VNE is 7% higher than MC-VNE in terms of the acceptance rate of virtual network requests, 43% higher than VNE-PSO, and 8% higher than MP-VNE.

In the following part, Fig. 8, Fig. 9 and Fig. 10 are all simulation results of DSCD-VNE in the simulation environment with the amount of virtual nodes as an unique variable. The purpose of these experiments are to verify that as the scale of virtual network increases, the trend of the indexes of the algorithms.

Experiment 3: The delay performance varies with the virtual network size

As can be seen from Fig. 8, with the expansion of virtual network size, the time delay of the overall virtual network has

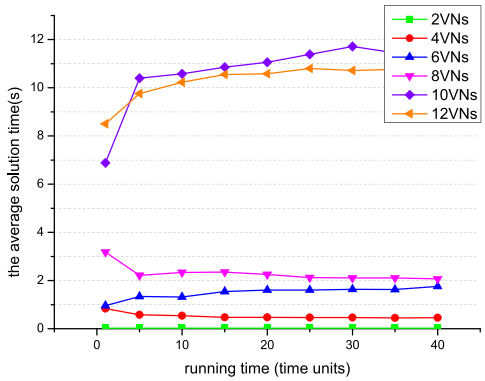


Fig. 9. Average solution time of different virtual network size.

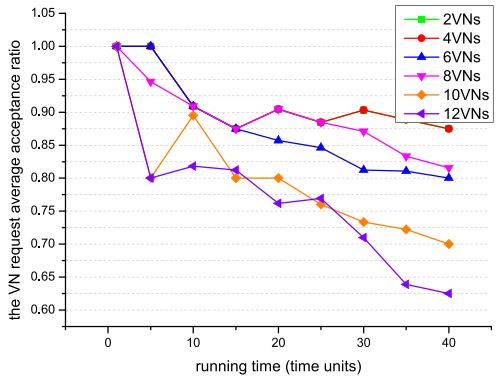


Fig. 10. The acceptance rate of virtual network request of different virtual network size.

been increased accordingly. However, due to the introduction of suitable algorithm for selecting the set of candidate substrate nodes and path splitting mechanism, the time delay optimization performance of the algorithm does not change significantly with time goes by and it is very stable. This simulation experiment proves the stability of the algorithm in the face of a large number of virtual network requests.

Experiment 4: Algorithm average solution time varies with the virtual network size

As shown in Fig. 9, for very small virtual network requests, the solution time of the algorithm can reach the level of milliseconds. For medium size virtual network requests, the algorithm can realize fast solution, and can basically enable the operator to bring the service online quickly within 5 seconds. Under the condition that the delay performance and the acceptance rate of the algorithm are acceptable, the introduction of MaxMatch and MaxMatched parameters is of great help to shorten the average solution time.

Experiment 5: Virtual network request acceptance rate varies with the virtual network size

As shown in Fig. 10, part of the curves overlap due to the pseudo-random number of the software, but this does not prevent our observation of the trend. As time goes by, the acceptance rate of virtual network requests will gradually decline, as the size of the virtual network increases, the acceptance rate will be lower. It can be seen that for medium-sized virtual network requests, the acceptance rate of virtual network requests can still maintain a relatively stable and gentle trend, and maintain above 80 percent.

From the above experimental results, we can conclude that DSCD-VNE performs better than VNE-PSO, MC-VNE and MP-VNE in terms of time delay and virtual network request acceptance rate. Not only in the specific numerical value, DSCD-VNE's time delay is shorter, the virtual network request acceptance rate is higher, with the passing of time, and the increase of hosting virtual network, its performance will not have a particularly obvious deterioration, it is more suitable for the actual virtual network mapping scene, our experiments fully proved its effectiveness.

In the algorithm design of VNE-PSO and MP-VNE, the link delay optimization in the network is not good enough due to node mapping first and link mapping next. Meanwhile, the performance of the discrete particle swarm optimization algorithm depends on the large number of particles and the high number of iterations, so when the solution time is acceptable or higher requirements are put forward, the algorithm performance will have a serious decline.

In the algorithm design of MC-VNE, link delay is given priority, so the delay performance of the network is better than that of VNE-PSO. However, in the selection stage of candidate nodes, there is a lack of sufficient consideration of the overall network resources, especially link resources, In addition, the link mapping stage lacks the processing of insufficient link resources, so the request acceptance rate of virtual network still needs to be improved.

In the algorithm design of DSCD-VNE, some problems existing in MC-VNE are fully considered. NRM and path splitting mechanism are introduced to reduce the debris of substrate network generated by virtual network embedding and enhance the acceptance rate of virtual network request. In the meantime, the influence of link delay on network delay is considered.

VII. Conclusion

In order to meet the requirements of low delay in multi-domain virtual network embedding, a delay-sensitive cross-domain virtual network mapping algorithm is proposed. The three-stage mapping algorithm of "node-link-node" proposed in this paper, as well as the referenced node resource measurement mechanism, path segmentation and k-shortest path algorithm, significantly optimize the delay performance of virtual network mapping, and make an important contribution to realize fast online service and improve the success rate of virtual network requests. The feasibility of this method is verified by a series of simulation experiments.

However, on the one hand, in the real cross-domain mapping scenario, due to the selfishness of InPs, they are not willing to disclose the detailed information about their network topology and resource availability to the third party, therefore, the performance of the algorithm cannot be well verified. On the other hand, network delay includes not only link delay, but also node forwarding delay, especially in LAN, node forwarding delay

has a great impact on the whole network delay. In the future work, we should not only consider the incomplete exposure of InPs, but also the forwarding delay of nodes.

REFERENCES

[1] I. Khan *et al.*, "A data annotation architecture for semantic applications in virtualized wireless sensor networks," *IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, Ottawa, ON, pp. 27–35, 2015, doi: 10.1109/INM.2015.7140273.

[2] K. A. Torkura and C. Meinel, "Towards vulnerability assessment as a service in openstack clouds," in *Proc. 41st Conf. Local Comput. Netw. Workshops (LCN Workshops)*, Dubai, 2016, pp. 1-8, doi: 10.1109/LCN.2016.022.

[3] S. Hong *et al.*, "Virtual optical network embedding in multi-domain optical networks," in *Proc. Commun. Conf.*, Austin, TX, 2014, pp. 2042–2047, doi: 10.1109/GLOCOM.2014.7037108.

[4] I. Houidi, W. Louati, W. Ben-Ameur, and D. Zeghlache, "Virtual network provisioning across multiple substrate networks," *Comput. Netw.*, vol. 55, no. 4, pp. 1011–1023, 2011.

[5] H. Cui, W. Gao, L. Jiang, and Y. Liu, "A virtual network embedding algorithm based on virtual topology connection feature," *16th Int. Symp. Wireless Pers. Multimedia Commun. (WPMC)*, pp. 1–5, Atlantic City, NJ, 2013.

[6] M. R. Rahman and R. Boutaba, "SVNE: survivable virtual network embedding algorithms for network virtualization," *IEEE Trans. Netw. Service Manage.*, vol. 10, no. 2, pp. 105–118, Jun. 2013.

[7] X. Wang, Q. Chen, and H. Qiu, "A effective two-step strategy of multi-domain virtual network embedding in 5g network slicing," in *Proc. The 3rd IEEE Int. Conf. Comput. Commun.*, 2017, pp. 1174–1179.

[8] P. Zhang, H. Li, Y. Ni, F. Gong, and F. Wang, "Security aware virtual network embedding algorithm using information entropy topsis," *J. Netw. Syst. Manage.*, vol. 28, no. 1, pp. 35–57, 2020.

[9] J. Ding, J. Liu, and Y. J. Liu, "Virtual network embedding for multi-topology virual network request," *J. Beijing Univ. Posts Telecommun.*, pp. 88–93, 2015.

[10] C. Yu, W. Hou, Y. Guan, Y. Zong, and P. Guo, "Virtual 5G network embedding in a heterogeneous and multi-domain network infrastructure," in *China Commun.*, vol. 13, no. 10, pp. 29–43, Oct. 2016, doi: 10.1109/CC.2016.7732010.

[11] L. Hui *et al.*, "Dynamic virtual network embedding over multilayer optical networks," *IEEE/OSA J. Opt. Commun. Netw.*, vol. 7, no. 9, pp. 918–927, Sep. 2015.

[12] D. Yun and Y. Yi, "Virtual network embedding in wireless multihop networks," in *Proc. Int. Conf. Future Internet Technologies*, 2011, pp. 30–33.

[13] H. Yu, T. Wen, H. Di, V. Anand, and L. Li, "Cost efficient virtual network mapping across multiple domains with joint intra-domain and inter-domain mapping," *Opt. Switching Netw.*, vol. 14, pp. 233–240, 2014.

[14] P. Zhang, "Incorporating energy and load balance into virtual network embedding process," *Comput. Commun.*, vol. 129, pp. 80–88, 2018.

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

[16] 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.

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

[18] Y. Li, Y. Jiang, D. Tian, L. Hu, H. Lu, and Z. Yuan, "Ai-enabled emotion communication," *IEEE Netw.*, vol. 33, no. 6, pp. 15–21, Nov.\Dec. 2019.

[19] H. Lu, Q. Liu, D. Tian, Y. Li, H. Kim, and S. Serikawa, "The cognitive internet of vehicles for autonomous driving," *IEEE Netw.*, vol. 33, no. 3, pp. 65–73, May\Jun. 2019.

[20] A. Fischer, J. F. Botero, M. T. Beck, and H. De Meer, "Virtual network embedding: A survey," *IEEE Commun. Surv. Tut.*, vol. 15, no. 4, pp. 1888–1906, Oct.\Dec. 2013.

[21] I. Houidi, W. Louati, and D. Zeghlache, "A distributed virtual network mapping algorithm," in *Proc. IEEE Int. Conf. Commun.*, Beijing, China, 2008, pp. 5634–5640, doi: 10.1109/ICC.2008.1056.

[22] L. Zhu, X. Tang, M. Shen, X. Du, and M. Guizani, "Privacy-preserving ddos attack detection using cross-domain traffic in software defined networks," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 628–643, Mar. 2018.

[23] S. Yang, F. Li, M. Shen, X. Chen, X. Fu, and Y. Wang, "Cloudlet placement and task allocation in mobile edge computing," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5853–5863, Jun. 2019.

[24] H. Yao, X. Yuan, P. Zhang, J. Wang, C. Jiang, and M. Guizani, "Machine learning aided load balance routing scheme considering queue utilization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7987–7999, Aug. 2019.

[25] H. Yao, P. Gao, J. Wang, P. Zhang, C. Jiang, and Z. Han, "Capsule network assisted iot traffic classification mechanism for smart cities," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7515–7525, Oct. 2019.

[26] H. Yao, D. Fu, P. Zhang, M. Li, and Y. Liu, "Msml: A novel multi-level semi-supervised machine learning framework for intrusion detection system," *IEEE Internet Things Journal*, vol. 6, no. 2, pp. 1949–1959, Apr. 2019.

[27] H. Yao, B. Zhang, P. Zhang, S. Wu, C. Jiang, and S. Guo, "RDAM: A reinforcement learning based dynamic attribute matrix representation for virtual network embedding," *IEEE Trans. Emerg. Topics Comput.*, to be published, doi: 10.1109/TETC.2018.2871549.

[28] H. Yao, T. Mai, J. Wang, Z. Ji, C. Jiang, and Y. Qian, "Resource trading in blockchain-based industrial internet of things," *IEEE Trans. Ind. Inform.*, vol.15, no.6, pp. 3602–3609, 2019.

[29] S. Saharan, S. Bawa, and N. Kumar, "Dynamic pricing techniques for intelligent transportation system in smart cities: A systematic review," *Comput. Commun.*, vol. 150, pp. 603–625, 2020.

[30] R. Rahman, T. Vinkó, D. Hales, J. A. Pouwelse, and H. J. Sips, "Design space analysis for modeling incentives in distributed systems," in *Proc. ACM SIGCOMM Conf. Appl., Technologies, Architectures, Protocols Comput. Commun.*, S. Keshav, J. Liebeherr, J. W. Byers, and J. C. Mogul, Eds., Toronto, ON, Canada, 2011, pp. 182–193.

[31] D. He, N. Kumar, M. K. Khan, L. Wang, and J. Shen, "Efficient privacy-aware authentication scheme for mobile cloud computing services," *IEEE Syst. J.*, vol. 12, no. 2, pp. 1621–1631, Jun. 2018.

[32] F. Samuel, M. Chowdhury, and R. Boutaba, "Polyvine: policy-based virtual network embedding across multiple domains," *J. Internet Services Appl.*, vol. 4, no. 1, pp. 1–23, 2013.

[33] D. Dietrich, A. Rizk, and P. Papadimitriou, "Multi-provider virtual network embedding with limited information disclosure," *IEEE Trans. Netw. Service Manage.*, vol. 12, no. 2, pp. 188–201, Jun. 2015.

[34] F. Bianchi and F. L. Presti, "A Markov Reward based Resource-Latency Aware Heuristic for the Virtual Network Embedding Problem," *SIGMETRICS Perform*, Evaluation Rev., vol. 44, no. 4, pp. 57–68, 2017.

[35] J. Yiming, M. Hailong, B. Youjun, S. Juan, and H. Lei, "Inter-domain virtual network embedding policy for revenue maximization," *Adv. Eng. Sci.*, vol. 50, no. 2, pp. 118–125, 2018.

[36] P. Li, "A multi-domain virtual network embedding algorithm based on minimum cost," *J. South China Univ. Technol. (Natural Science Edition)*, no. 9, vol. 9, pp. 67–73, 2015.

[37] G. S. Aujla, R. Chaudhary, S. Garg, N. Kumar, and J. J. Rodrigues, "Sdn-enabled multi-attribute-based secure communication for smart grid in iiot environment," *IEEE Trans. Ind. Inform.*, vol. 14, no. 6, pp. 2629–2640, Jun. 2018.

[38] 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)*, Split, 2016, pp. 1–6, doi: 10.1109/SOFTCOM.2016.7772158.

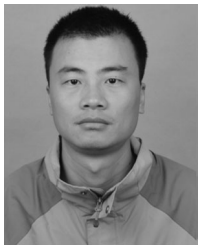
[39] X. Xiao, X. Zheng, and Y. Zhang, "A multidomain survivable virtual network mapping algorithm," *Secur. Commun. Netw.*, vol. 2017, pp. 5258010:1–5258010:12, 2017.

[40] W. Xiaolei, C. Qiang, and L. Caixia, "Cross-domain virtual network mapping based on discrete particle swarm optimization and kruskal algorithm in 5g network slicing," *Appl. Res. Comput.*, vol. 36, no. 4, pp. 1169–1173, 2019.

[41] M. M. Hasan, H. Amarasinghe, and A. Karmouch, "Network virtualization: Dealing with multiple infrastructure providers," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Ottawa, ON, Canada, 2012, pp. 5890–5895, doi: 10.1109/ICC.2012.6364756.

[42] P. Gope, R. Amin, S. Hafizul Islam, N. Kumar, and V. K. Bhalla, "Lightweight and privacy-preserving rfid authentication scheme for distributed iot infrastructure with secure localization services for smart city environment," *Future Gener. Comput. Syst.*, vol. 83, pp. 629–637, 2018.

[43] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *Proc. IEEE INFOCOM '96. Conf. Comput. Commun.*, vol. 2, pp. 594–602, Mar. 1996.



Peiying Zhang is currently an Associate Professor in the College of Computer Science and Technology from China University of Petroleum (East China). He received the Ph.D. degree in the School of Information and Communication Engineering at the University of Beijing University of Posts and Telecommunications in 2019. His research interests include future internet architecture, network virtualization, and artificial intelligence for networking. He has published more than 50 papers in prestigious peer-reviewed journals and conferences.

Xue Pang is currently a graduate student in the College of Computer Science and Technology, China University of Petroleum (East China). Her research interests include network virtualization and artificial intelligence for networking.

Yanxian Bi is currently an Engineer in the National Engineering Laboratory for Public Safety Risk Perception and Control by Big Data (PSRPC). He received his Ph.D. degree from the University of Beihang University in 2017. During 2015–2017, he studied in the University of Birmingham as a Visiting Scholar. His research interests focus on deep learning and computational intelligence.

Haipeng Yao (Member, IEEE) is an Associate Professor in the Beijing University of Posts and Telecommunications. Haipeng Yao received his Ph.D. in the Department of Telecommunication Engineering at the University of Beijing University of Posts and Telecommunications in 2011. He has been engaged in research on future internet architecture, network AI, Big Data, cognitive radio networks, and optimization of protocols and architectures for broadband wireless networks. He has published more than 80 papers in prestigious peer-reviewed journals and conferences.

Huijiang Pan is currently pursuing the master's degree in information and communication engineering at the State Key Laboratory of Networking and Switching Technology, the Beijing University of Posts and Telecommunications, Beijing, China. His current research interests include network virtualization, artificial intelligence operations, and machine learning.

Neeraj Kumar (Senior Member, IEEE) received the Ph.D. degree in computer science and engineering from Shri Mata Vaishno Devi University, Katra, India. He is currently with the Department of Computer Science and Engineering, Thapar University, Patiala, India. He has authored or coauthored more than 400 technical research papers in leading journals such as the IEEE TII, IEEE TIE, IEEE TDSC, IEEE TWPS, IEEE SYSTEMS JOURNAL, *IEEE COMMUNICATIONS MAGAZINE, IEEE WIRELESS COMMUNICATIONS MAGAZINE, IEEE NETWORK MAGAZINE*, and conferences. His research interests include mobile computing, parallel/distributed computing, multiagent systems, service-oriented computing, routing and security issues in mobile ad hoc, and sensor and mesh networks. He is Associate Editor of ACM Computing, Survey, IEEE SYTEMS JOURNAL, IEEE TRANSACTION ON SUSTAINABLE COMPUTING, *Elsevier JNCA, Elsevier Comcom, IEEE Network Magazine, IEEE Communication Magazine, Wiley IJCS*. He has been in the list of highly cited researcher of 2019 list of WoS. He has edited special issues of many top-tiered journals and conducted workshops in IEEE ICC, IEEE Glboecom. He has won many prestigious awards from IEEE.