Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 5452397, 17 pages
https://doi.org/10.1155/2020/5452397

Research Article

Novel Node-Ranking Approach for SDN-Based Virtual Network Embedding

Chaowei Shi, Xiangru Meng, Qiaoyan Kang, and Xiaoyang Han

College of Information and Navigation, Air Force Engineering University, Xi'an, Shaanxi 710077, China

Correspondence should be addressed to Chaowei Shi; cwshi0839@163.com

Received 16 July 2020; Revised 14 September 2020; Accepted 30 September 2020; Published 17 October 2020

Academic Editor: Neale R. Smith

Network virtualization is considered as a key technology for the future network. The emergence of software-defined network (SDN) provides a platform for the research and development of network virtualization. One of the key challenges in network virtualization is virtual network embedding (VNE). Some of the previous VNE algorithms perform virtual node embedding, which combines the nodes' resource attributes and local topology attributes by arithmetic operations. On the one hand, it is not easy to distinguish the topological differences between SN and VN only by simple topology metrics. On the other hand, it is easy to ignore the different weight impacts of different metrics using only arithmetic operations, which will lead to an unbalanced embedding solution. To deal with these issues, we propose a novel node-ranking approach based on topology-differentiating (VNE-NRTD) for SDN-based virtual network embedding. Owing to the topological difference between SN and VN, different node metrics are used to quantify the substrate nodes and virtual nodes, respectively. Then, the nodes are ranked using the modified set pair analysis (SPA) method to avoid the unbalanced embedding solution. On this basis, we introduce the global bandwidth of the network topology into node-ranking to further improve the efficiency of node embedding. The simulation results show that the VNE-NRTD algorithm proposed in this paper outperforms other latest heuristic algorithms in terms of the VNR acceptance ratio, long-term average R/C ratio, substrate node utilization, and substrate link utilization.

1. Introduction

Over the past decades, Internet has achieved great success. However, the ossification of Internet is getting worse with the rapid growth of end-users and various applications. Inspired by the success of virtualization technology in the field of computing [1], network virtualization (NV) [2] allows multiple heterogeneous virtual networks (VNs) to run on a shared substrate network (SN) simultaneously [3] and is gradually becoming an important research topic in academia and industry recently.

In the NV environments, traditional Internet service providers (ISPs) are decoupled into infrastructure providers (InPs) and service providers (SPs) [4]. The InPs are responsible for the operation and maintenance of the substrate network infrastructures. The SPs construct different VNs according to the user's service request and rent SN resources from the InPs to fulfill these requirements, which is called virtual network embedding (VNE) [5] in academia.

VNE is a process of embedding the virtual network requests (VNRs) onto the shared SN infrastructures, with the constraints of nodes (e.g., node computing capacity and node location) and links (link bandwidth). This makes the VNE be an NP-hard problem [2]; hence, a considerable volume of algorithms has been proposed to solve the VNE problem.

Some studies addressed this problem with exact algorithms [7, 8]. Although it can get better results after multiple optimization computations, its computational complexity grows exponentially with the increases in the network scale [9]. Therefore, it is unsuitable to adopt exact algorithms to embed VNs for large-scale networks. Other exact-like algorithms [10] have to relax or cut down the integer variable constraints to achieve a workable VNE in a limited time.

Besides, numerous heuristic algorithms [11] have been proposed to find the near-optimal solution of VNE. Part of studies [12, 13] used one-stage heuristic embedding algorithms to solve the VNE problem. The drawback of these

algorithms is that it can easily lead to multiple backtracking, which affects the performance of the algorithm. Therefore, a large number of two-stage heuristic VNE algorithms [6, 14–18] have been designed over the past years, which divide VNE into two stages: (1) virtual node embedding stage and (2) virtual link embedding stage. After the nodes are successfully embedded through the greedy strategy, Dijkstra's algorithm [19], *k*-shortest path algorithm [20], or multicommodity flow model [21] are used to search a loop-free path between the corresponding substrate nodes for virtual link embedding.

Most of these algorithms perform node embedding only to combine the nodes' resource attributes (e.g., nodes' computing capability and their adjacent link bandwidths) and local topology attributes (e.g., node degree and node centrality) by simple arithmetic operations, while without fully considering the weight of different node metrics in different network environments. This would lead to the unbalance problem of these metrics and reduce the resource utilization of SN.

Moreover, only the resource and topology attributes of the nodes to be embedded are considered in the node-ranking process. The attributes of its adjacent nodes and the importance of the node in the global network topology are not considered. This can easily give rise to poor node-ranking results. The other two latest heuristics consider global network resource [22, 23] during the node embedding stage. However, the global network resource metrics considered are oversimplified.

On the contrary, we all know that the goal of VNE is to make full use of SN resources and improve the revenue of InPs. Therefore, it is also an important issue to efficiently adjust the entire network resources. However, the distributed architecture of the traditional networks makes it not easy for InPs to obtain global network resources in real-time and dynamically manage network resources.

The emergence of the software-defined network (SDN) [24] has provided powerful convenience for the implementation and application of network virtualization, while VNE has become a heavyweight application in SDN applications. Since the SDN controller has an outstanding ability to call and control global resource, it is possible to achieve reasonable scheduling of global resource on the substrate network and improve the efficiency of resource allocation through the SDN controller.

To describe the VNE problem more clearly, this paper studies the model of SDN-based VNE and lists the mathematical formulation in Section 3. VNs can be abstracted as applications for users deployed on the substrate network, which can apply for network resources from the SDN controller through the northbound interface. SDN controller can grasp the dynamic changes in the SN resources and regulate the global resource of the SN at the same time. The substrate switches forward the payload data according to the contents of the flow rule issued by the SDN controller through the southbound interface.

Furthermore, in view of the low request acceptance ratio and high embedding cost in VNE, a novel node-ranking approach based on topology-differentiating is proposed for virtual network embedding. The main difference between our paper and previous works is that we take into consideration the topology difference between SN and VN and use different node metrics to quantify the substrate nodes and virtual nodes, respectively. Then, different from the previous node-ranking methods that ignore the weight difference between different metrics, the modified set pair analysis (SPA) [25] method is adopted to rank all virtual nodes and substrate nodes in advance, which can assign appropriate weights to different metrics according to the network environment. Corresponding subalgorithm is labeled as VNE-NRTD-S. Finally, the global bandwidth is introduced into the iterative operation to enhance node-ranking reliability and improve the resource utilization of SN. Corresponding subalgorithm is labeled as VNE-NRTD-G.

To verify the efficiency of the proposed algorithm, a comprehensive comparison of algorithm performance is performed in this paper. The results validate that the VNE-NRTD algorithm proposed in this paper performs better than the latest three algorithms.

Overall, the main contributions of this paper are summarized as follows:

(i) To describe the VNE problem more clearly, this paper studies the virtual network embedding model based on SDN. The mixed-integer linear programming (MILP) model is established for the SDN-based virtual network embedding problem, and the flow rule capacity constraint is added in the node embedding process.

(ii) A two-stage heuristic algorithm VNE-NRTD is proposed based on topology-differentiating to find the near-optimal solution of MILP. According to the topology difference between SN and VN, different node metrics are used to quantify the substrate nodes and virtual nodes, respectively. Different from the previous node-ranking methods that ignore the weight difference between different metrics, an appropriate weight is assigned to each metric and the node importance is calculated based on the modified SPA method. On this basis, the global bandwidth of the network topology is introduced into the node-ranking process to further improve the efficiency of node embedding.

(iii) Comprehensive performance evaluation is performed to verify the efficiency of the VNE-NRTD algorithm. Three typical and latest heuristic algorithms are selected for performance comparison. Simulation results vividly demonstrated that the proposed VNE-NRTD algorithm has obvious improvements compared with the selected algorithm in terms of the VNR acceptance ratio, long-term average R/C ratio, and substrate node utilization under different network environments, while reducing the substrate link utilization.

The remainder of this paper is organized as follows: Section 2 briefly reviews the related work of VNE; the VNE model based on SDN is discussed in Section 3; in Section 4,

we present the MILP model for VNE, and the heuristic VNE-NRTD algorithm is detailed in Section 5; extensive simulation work is implemented in Section 6; and Section 7 concludes this paper.

2. Related Work

VNE is formulated as a resource allocation problem, which is NP-hard. Existing VNE algorithms can be roughly divided into three categories: (i) the exact and exact-like algorithms, (ii) the metaheuristic algorithms, and (iii) the heuristic algorithms. In this section, we briefly look back on some previous and latest studies in terms of these three aspects.

2.1. Exact and Exact-Like VNE Algorithms. The exact algorithms solve the VNE problem based on the fixed mathematical programming model, to different goals, and the exact embedding of each given VN is different [8].

Chowdhury et al. [26] first formulated the VNE as a mixed-integer program (MIP) problem through substrate network augmentation. Owing to the computational complexity of the MIP model, they relaxed the integer constraints and turned the MIP model to a linear program (LP) model. VNRs are then embedded by using deterministic and random rounding techniques. Hu et al. [7] enhanced the path-based MIP model formulated based on previous research. They developed an iterative process based on the primal-dual analysis of the MIP model, which allowed feedbacks between the link embedding subproblem and the node assignment subproblem. After multiple iterations, their decomposition approach can lead to a near-optimal solution to the VNE problem.

Cao et al. [10] proposed an efficient VNE algorithm based on restrictive selection and optimization theory. The algorithm can calculate the suboptimal embedding in limited time by largely cutting down on the number of integer variables. The restrictive selection contributes to selecting candidate substrate nodes and paths.

Farkiani et al. [27] used the stochastic programming framework to formulate the stochastic VNE problem for the first time. The authors used the Benders method and decomposed the stochastic VNE problem into two tractable subproblems to improve the scalability of the technique. Then, they proposed a novel iterative algorithm to reduce the problem-solving time via new acceleration techniques applied on the Benders method.

The constraints of these exact algorithms are only node capacity and link bandwidth. The latest exact algorithms introduced new constraints to extend the VNE problem to a more practical online VNE process. Yang et al. [28] employed an exact VNE algorithm, which is based on the integer linear program (ILP) model, for embedding virtual network requests with location constraints. Lu et al. [29] presented a novel collaborative VNE algorithm based on the LP model, which could bring about the coordinative embedding between different VNRs.

2.2. Metaheuristic Algorithms. Given that the exact VNE algorithms would consume a large amount of computation time, some scholars have introduced metaheuristic algorithms into VNE, which can reduce the running time of the algorithm while ensuring the efficiency of VNE. These types of algorithms are an alternative way to obtain near-optimal solutions at a limited time, especially for NP-hard problems. In this part, we will briefly introduce the application of metaheuristic algorithms.

Fajjari et al. [30] employed the ant colony optimization algorithm to deal with the VNE problem, which aims to minimize the reject rate of requests and maximize the revenue of InPs. The near-optimal solution can be found by iteratively improving candidate solutions. Zhu et al. [31] proposed a modified ant colony optimization algorithm to solve the VNE problem. The combination of revenue and acceptance ratio of an SN is used as an important component when designing the fitness function to evaluate iterative solutions obtained by ants, and the pheromone update rules are designed based on the fitness function.

Pathak et al. [32] exploited the basic genetic algorithm for VNE problem to embed multiple VNRs on InPs that manage multiple SNs. However, the limitations of the genetic algorithm make the problem easy to early into the local optimal solution, and the convergence speed is slow. Liu et al. [33] established a new VNE algorithm based on mixed genetic algorithm, which regarded VNE as a MILP model to maximize the revenue of InPs. The simplex method is used to estimate the optimal direction. The proposed algorithm took advantage of the genetic algorithm and simplex method to optimize the embedding scheme, as much as possible to avoid local optimal. Zhang et al. [5] proposed an approach of VNE based on modified genetic algorithm. This algorithm considered the bandwidth utilization of link embedding as the optimization target and performed the VNE process through selection, crossover, mutation, and feasibility checking operations.

Although metaheuristic algorithms can be used to find near-optimal solutions for large instances by iteratively improving a candidate solution, these algorithms are highly dependent on the initial population and the embedding results are relatively random. We intend to combine this work with deep learning and artificial intelligence in our future work.

2.3. Heuristic Algorithms. Owing to the limitations of exact algorithms and metaheuristic algorithms, some works have to design a large number of two-stage heuristic VNE algorithms, which consider the resource and topology information of the nodes more or less.

Yu et al. [21] ranked the nodes based on local resource attributes and used a greedy algorithm to embed the virtual nodes. Wang et al. [14] considered the topology information of substrate nodes and virtual nodes, and they ranked the nodes based on its centrality. Cui et al. [15] considered the proximity of the substrate nodes corresponding to the embedded virtual nodes when ranking the substrate nodes, so as to avoid blindly calculating the global centrality of the

nodes. Liu [17] et al. introduced the node topology potential to find the optimal substrate node based on the field theory, which improved the poor match between the virtual nodes and the substrate nodes.

However, these algorithms perform node embedding only to combine the node's resource attributes and local topology attributes by simple arithmetic operations, without fully considering the weight of different node metrics in different network environments. In response to this problem, Gong et al. [16] used the Technique for Order Preference by Similarity to an Ideal Solution (TOPSIS) for the first time to rank the nodes with multiple metrics. Shooshtarian et al. [18] applied the TOPSIS method to the node ranking in the process of secure VNE. Nevertheless, neither of them considered the case where the points on the vertical line are in the same distance from the positive and negative ideals.

The latest VNE algorithms extend the VNE problem to a more realistic approach for more accurate and realistic online VNE processes. Chen et al. [6]. proposed a novel VNE model in SDN and P4 [34] environments that considered the label, computing, and bandwidth constraints. It generated virtual network identifiers by combining the link-grained labels with their location information. Since the virtual links are embedded onto one or more substrate links that may be connected through intermediate substrate nodes, the resource required by the intermediate substrate node to forward the data traffic is also considered in [35]. Zhao et al. [36] focused on embedding the virtual networks onto the distributed soft-defined network with multiple controllers. Zhang et al. [37] considered the storage resource parameter of the substrate nodes for the first time in the virtual network embedding, which is similar to the parameter of flow rule capacity in the SDN environment.

2.4. Innovations to Existing Two-Stage Heuristic Algorithms. The VNE-NRTD algorithm proposed in this paper differs from previous heuristic algorithms in four main aspects.

First of all, to extend the VNE problem to a more realistic approach, we study the virtual network embedding model based on SDN. Then, according to the topological difference between SN and VN, different node metrics are used to quantify the substrate nodes and virtual nodes, respectively.

Third, the importance of nodes is calculated based on the modified SPA method. On this basis, the global bandwidth of the network topology is introduced into the node-ranking process to further improve the efficiency of node embedding.

Finally, we perform a comprehensive performance evaluation against typical and latest heuristic algorithms.

3. SDN-Based VNE Model

This section describes the VNE process based on SDN, including the SDN-based VNE model, SN resources measurement, VN embedding stages, and main performance evaluation metrics.

3.1. VNE Model Based on SDN. Abstract modeling of SN and VN is needed in the research of VNE, so as to discuss the constraints of resource and location in the embedding process. In this section, we layer the SDN-based VNE model to discuss the hardware components and functions performed by each layer. We abstract this model into the following four layers and illustrate it in Figure 1.

3.1.1. User Layer. At the user layer, different tenants or applications, which can adopt different architectures and carry out different network protocols, request network resources from SPs to fulfill their functional requirements.

3.1.2. Service Provide Layer. The SPs construct the logical VN topology according to the requirements of end user and rent substrate network resources from the SDN controller through the northbound interface. The VN constructed by SPs is a logical topology consisting of nodes and links, which can be abstracted as an undirected graph $G^v = (N^v, L^v)$; therein, N^v represents the set of virtual nodes and L^v represents the set of virtual links. Each virtual node $n^v \in N^v$ is characterized by its required flow rule capacity $(\mathrm{Flow}(n^v))$ and computing capability $(\mathrm{Cpu}(n^v))$. Each virtual link $l^v \in L^v$ has a required bandwidth $\mathrm{Bw}(l^v)$. A VNR can be defined as $\mathrm{VNR}_i = (G^v, D^v, T^v)$, where G^v indicates the VN topology and D^v is the deviation relationship between virtual node $n^v \in N^v$ and any selected substrate nodes. $T^v = (t_a, t_{\ d})$ indicates the duration time of the VNR, where t_a indicates the arrival time of the VNR and $t_{\ d}$ indicates the end time of the VNR.

The upper part of Figure 1 illustrates two VNRs with different topologies constructed by SPs, where *a*, *b*, *c*, *d*, and *e* are virtual nodes. The number in the left part of the box represents the computing capability demand. The number in the right part of the box denotes the flow rule capacity demand. The number over the link denotes the required link bandwidth. For simplicity, the location constraints of virtual nodes and the time attributes of VNRs are omitted in this figure.

3.1.3. Control Layer. According to [8, 38], it can be understood that FlowN adopts a logical distributed controller method, and each VN has a logical controller that can independently call and control the network resources requested by itself. Therefore, this paper chooses FlowN as the virtualization platform for VNE. On this platform, the SDN controller executes the resource scheduling algorithm, that is, the VNE-NRTD algorithm. In the control layer, the SDN controller combines the attributes of the underlying SN resources to perform resource scheduling on the OpenFlow (OF) switches in the data layer and efficiently allocates substrate network resources to each virtual network.

3.1.4. Data Layer. The data layer is one or more SNs that consist of multiple OF switches with the data forwarding function. OF switches perform traffic forwarding according to the contents of the flow rules issued by the controller. In

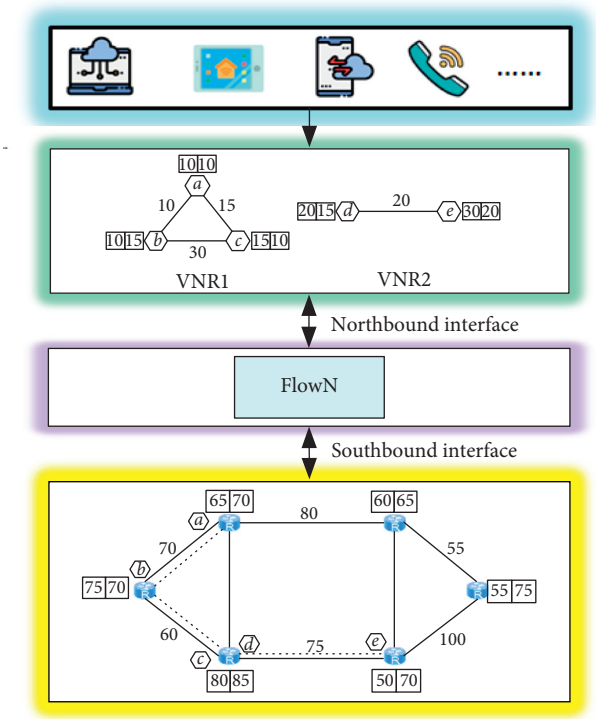

Figure 1: VNE model based on SDN.

SDN architecture, the substrate network also can be abstracted as an undirected graph $G^s = (N^s, L^s)$, where N^s represents the set of substrate nodes (bottom switches) and L^s represents the set of substrate links. Each substrate node is associated with its remaining computing capability $(R^s_{\mathrm{Cpu}}(n^s))$, flow rule capacity $(R^s_{\mathrm{Flow}}(n^s))$, and location coordinates $(\mathrm{loc}(n^s))$ that are denoted by x and y coordinates. The basic attribute of each substrate link is its remaining bandwidth $(R^s_{\mathrm{Bw}}(l^s))$.

The lower part of Figure 1 illustrates a substrate network model in the control area, where A, B, C, D, E, and F are substrate nodes. The number in the left part of the box indicates the remaining computing resources of the node. The number in the right part of the box indicates the remaining flow rule capacity of the nodes. The number over the link indicates the remaining link bandwidth. Likewise, the location of substrate nodes is not plotted either.

3.1.5. VNE Notations. The main notations used throughout this paper are listed in Table 1.

3.2. Measurement of Substrate Network Resources. In the SDN-based virtual network embedding area, the SDN controller can grasp the dynamic changes in SN resources in real time, to quantify the resource usage of substrate nodes, and the remaining available computing capability of node M in SN is defined as follows:

$$R^s_{\mathrm{Cpu}}(M) = \mathrm{Cpu}(M) - \sum_{\forall n^v \uparrow M} \mathrm{Cpu}(n^v), \tag{1}$$

Table 1: VNE notations.

G^s	Substrate network
N^s	Set of substrate nodes
n^s, M, N	Substrate nodes
L^s	Set of substrate links
l^s, MN	Substrate links
$\mathrm{Cpu}(n^s)$	Total computing capability of substrate nodes
$R^s_{\mathrm{Cpu}}(n^s)$	Remaining node computing capability
$\mathrm{Flow}(n^s)$	Total flow rule capacity of substrate nodes
$R^s_{\mathrm{Flow}}(n^s)$	Remaining flow rule capacity of substrate nodes
$\mathrm{Bw}(l^s)$	Total bandwidth of substrate links
$R^s_{\mathrm{Bw}}(l^s)$	Remaining bandwidth of substrate links
G^v	Virtual network
N^v	Set of virtual nodes
n^v, m, n	Virtual nodes
L^v	Set of virtual links
l^v, mn	Virtual links
$\mathrm{Cpu}(n^v)$	Computing capability demand of virtual nodes
$\mathrm{Flow}(n^v)$	Flow rule capacity demand of virtual nodes
$\mathrm{Bw}(l^v)$	Bandwidth demand of virtual links

where $\mathrm{Cpu}(M)$ indicates the total computing capability of substrate nodes and $\forall n^v \uparrow M$ indicates that the virtual node n^v is embedded onto the substrate node M.

Similarly, the remaining available flow rule capacity of node M in SN is defined as follows:

$$R^s_{\mathrm{Flow}}(M) = \mathrm{Flow}(M) - \sum_{\forall n^v \uparrow M} \mathrm{Flow}(n^v). \tag{2}$$

The remaining available bandwidth of the link MN is defined as

$$R^s_{\mathrm{Bw}}(MN) = \mathrm{Bw}(MN) - \sum_{\forall l^v \uparrow MN} \mathrm{Bw}(l^v), \tag{3}$$

where $\forall l^v \uparrow MN$ indicates that the virtual link l^v is embedded onto the substrate link MN.

3.3. VNR Embedding. The embedding process of each VNR consists of the following two stages: (1) the stage dealing with the embedding of virtual nodes; (2) the stage of embedding the virtual links. The process of VNE is shown in Figure 2.

3.3.1. Node Embedding Stage. Each substrate node can only host one node in the same virtual network, while the nodes in different VNs can be embedded onto the same substrate node and are subject to

$$\mathrm{Cpu}(n^v) \le R^s_{\mathrm{Cpu}}(n^s), \tag{4}$$

$$\mathrm{Flow}(n^v) \le R^s_{\mathrm{Flow}}(n^s), \tag{5}$$

$$\mathrm{Dis}(\mathrm{loc}(n^s), \mathrm{loc}(n^v)) \le D, \tag{6}$$

where equations (4) and (5) make sure that substrate nodes have sufficient computing capability and flow rule capacity to host virtual nodes. The node location constraint is defined in equation (6), the allowed maximum deviation of virtual node n^v is D, and the deviation must be within the radius D

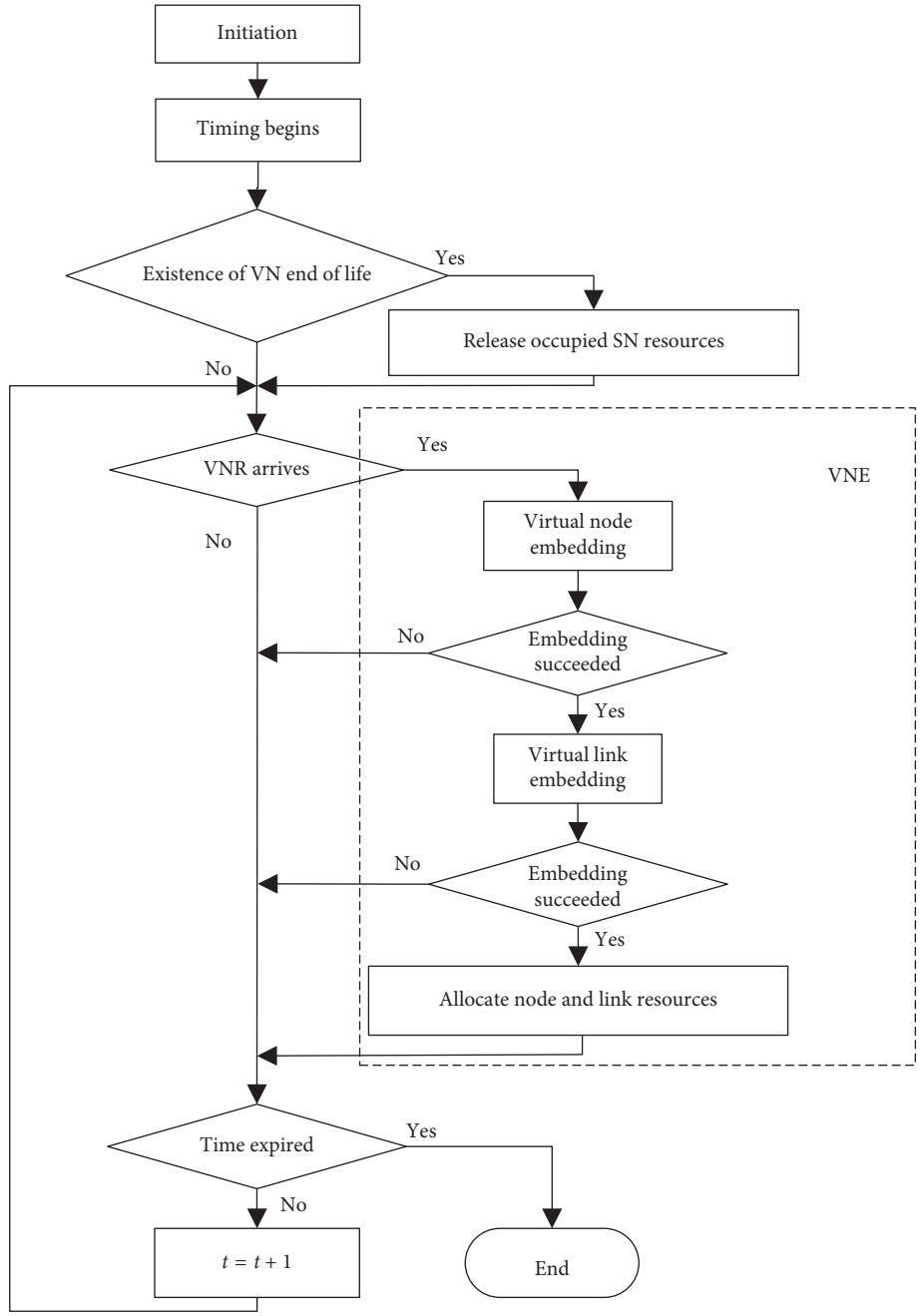

Figure 2: The process of VNE.

of virtual node n^v. The virtual node embedding results demonstrated in Figure 1 are $\{a \longrightarrow A, b \longrightarrow C, c \longrightarrow E\}$ and $\{d \longrightarrow E, e \longrightarrow F\}$.

3.3.2. Link Embedding Stage. Each substrate link can host multiple virtual links in different VNs and is subject to

$$\mathrm{Bw}\left(l^v\right) \leq R^s_{\mathrm{Bw}}\left(l^s\right), \tag{7}$$

where equation (7) ensures that the remaining bandwidth of the substrate link is greater than the bandwidth demand of the virtual link. The virtual link embedding result demonstrated in Figure 1 is $\{(a, b) \longrightarrow (A, C), (a, c) \longrightarrow (A, E), (b, c) \longrightarrow (C, E)\}$ and $\{(d, e) \longrightarrow (E, F)\}$.

3.4. Performance Evaluation Metrics

3.4.1. VNR Acceptance Ratio. The acceptance ratio of VNR is defined in equation (8), and it indicates the ratio of the number of successfully embedded VNRs to the total number of VNRs in the continuous time T:

$$\omega_{\mathrm{accept}} = \lim_{T \longrightarrow \infty} \frac{\mathrm{NUM}_{\mathrm{vsuc}}(T)}{\mathrm{NUM}_v(T) + \delta}, \tag{8}$$

where $\mathrm{NUM}_{\mathrm{vsuc}}(T)$ and $\mathrm{NUM}_{v}(T)$ represent the number of VNRs successfully embedded and the total number of VNRs in time T, respectively. δ is an infinitely small positive number.

3.4.2. Long-Term Average R/C Ratio. For InPs, the purpose of VNE is to reduce the SN resources consumption while guaranteeing revenue. This paper uses the long-term average R/C ratio to evaluate the performance of the algorithm.

Similar to the work in [36], the revenue of InPs can be denoted as

$$\mathrm{Rev}\left(G^{v}\right) = \sum_{n^{v}\in N^{v}} \left(\alpha\,\mathrm{Cpu}\left(n^{v}\right) + \beta\,\mathrm{Flow}\left(n^{v}\right)\right) + \gamma \sum_{l^{v}\in L^{v}} \mathrm{Bw}\left(l^{v}\right), \tag{9}$$

where α, β, and γ represent the proportion of $\mathrm{Cpu}(n^{v})$, $\mathrm{Flow}(n^{v})$, and $\mathrm{Bw}(l^{v})$ resources in revenue and cost, respectively.

Likewise, we denote the cost of SPs as the total amount of substrate resources allocated to the VNR:

$$\mathrm{Cost}\left(G^{v}\right) = \sum_{n^{v}\in N^{v}} \left(\alpha\,\mathrm{Cpu}\left(n^{v}\right) + \beta\,\mathrm{Flow}\left(n^{v}\right)\right) + \gamma \sum_{l^{v}\in L^{v}} \mathrm{hops}\left(l^{v}\right)\mathrm{Bw}\left(l^{v}\right), \tag{10}$$

where $\mathrm{hops}(l^{v})$ is the number of hops in the substrate link corresponding to the virtual link.

The long-term average R/C ratio can be defined as follows:

$$\frac{\mathrm{Rev}}{\mathrm{Cost}} = \lim_{T\longrightarrow\infty} \frac{\sum_{t=0}^{T} \mathrm{Rev}\left(G^{v},t\right)}{\sum_{t=0}^{T} \mathrm{Cost}\left(G^{v},t\right)}. \tag{11}$$

4. Mixed-Integer Programming Formulation for VNE

In this paper, we formulate the MILP model intending to maximize the long-term average R/C ratio. The detailed description is presented as follows.

4.1. Objective Function. The objective function could be denoted as

$$\max\left\{\frac{\mathrm{Rev}}{\mathrm{Cost}} = \lim_{T\longrightarrow\infty} \frac{\sum_{t=0}^{T} \mathrm{Rev}\left(G^{v},t\right)}{\sum_{t=0}^{T} \mathrm{Cost}\left(G^{v},t\right)}\right\}. \tag{12}$$

4.2. VNE_MILP

4.2.1. Variables.

$$\forall m \in N^{v}, \forall M \in N^{s},$$
$$\Upsilon_{m}^{M} = \begin{cases} 1, & \text{if } m \text{ is embedded onto } M, \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

The binary variable Υ is used to represent the embedding relationship between the virtual node and the substrate node. In equation (13), if the virtual node m is embedded onto the substrate node M, $\Upsilon = 1$; otherwise, $\Upsilon = 0$.

$$\forall mn \in L^{v}, \forall MN \in L^{s},$$
$$\Gamma_{mn}^{MN} = \begin{cases} 1, & \text{if } mn \text{ is embedded onto } MN, \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

The binary variable Γ is used to represent the embedding relationship between the virtual link and the substrate link. In equation (14), if the virtual link mn is embedded onto the substrate link MN, $\Gamma_{mn}^{MN} = 1$; otherwise, $\Gamma_{mn}^{MN} = 0$.

4.2.2. Constraints of Node Embedding.

$$\forall m \in N^{v}, \forall M \in N^{s}, \Upsilon_{m}^{M}\mathrm{Cpu}\left(m\right) \le R_{\mathrm{Cpu}}\left(M\right), \tag{15}$$

$$\forall m \in N^{v}, \forall M \in N^{s}, \Upsilon_{m}^{M}\mathrm{Flow}\left(m\right) \le R_{\mathrm{Flow}}\left(M\right), \tag{16}$$

$$\Upsilon_{m}^{M}\mathrm{dis}\left(\mathrm{loc}\left(M\right),\mathrm{loc}\left(m\right)\right) \le D, \tag{17}$$

$$\forall M \in N^{s}, \sum_{m\in N^{v}} \Upsilon_{m}^{M} \le 1, \tag{18}$$

$$\forall m \in N^{v}, \sum_{M\in N^{s}} \Upsilon_{m}^{M} = 1. \tag{19}$$

Equation (15) ensures that the available computing capability of the candidate substrate node should be greater than the computing capability demand of the virtual node. Equation (16) ensures that the available flow rule capacity of the candidate substrate nodes should be greater than the flow rule capacity demand of the virtual node. Equation (17) is the location constraint, where $\mathrm{dis}(\mathrm{loc}(M),\mathrm{loc}(m))$ denotes the distance between the substrate node M and the virtual node m. Equation (18) ensures that the nodes in the same VN topology cannot be embedded onto the same substrate node. Equation (19) ensures that each virtual node can only be embedded onto at most in one substrate node.

4.2.3. Constraints of Link Embedding.

$$\sum_{MN\in L^s}\Gamma_{mn}^{MN}-\sum_{MN\in L^s}\Gamma_{mn}^{NM}=\begin{cases}1, & \Upsilon_m^N=1,\\ -1, & \Upsilon_m^M=1,\\ 0, & \text{otherwise},\end{cases}\quad \begin{aligned}&\forall M\in N^s,\\ &\forall mn\in L^v,\end{aligned} \tag{20}$$

$$\sum_{MN\in L^s}\Gamma_{mn}^{MN}\mathrm{Bw}(mn)\le \mathrm{Bw}(MN),\quad \begin{aligned}&\forall mn\in L^v,\\ &\exists MN\in L^s,\end{aligned} \tag{21}$$

$$\sum_{MN\in L^s}\Gamma_{mn}^{MN}\le 1.\quad \begin{aligned}&\forall mn\in L^v,\\ &\forall MN\in L^s,\end{aligned} \tag{22}$$

Equation (20) is the connectivity constraint. It refers to the flow conservation constraint for routing one unit of net flow from virtual node m to virtual node n. Equation (21) ensures that the available bandwidth of the candidate substrate link should be greater than the bandwidth demand of the virtual link. Equation (22) ensures that there are no path splits and each virtual link is embedded onto up to one substrate path.

5. Heuristic Algorithm Design

Owing to the high complexity of the exact algorithm, it is unsuitable for large-scale network topology environments. Therefore, we propose a novel heuristic algorithm called VNE-NRTD to find the near-optimal solution of MILP.

This section details the proposed VNE-NRTD algorithm. We first introduce the node-ranking metrics. Then, two novel node-ranking methods are introduced. Based on the results of node ranking, the greedy strategy is used for node embedding. After the nodes are embedded successfully, the shortest path algorithm is used to perform virtual link embedding. Finally, we analyze the algorithm complexity and validate that the VNE-NRTD algorithm proposed in this paper is polynomial time-solvable.

5.1. Node-Ranking Metrics. Similar to [8], the importance of node n is measured by the node's computing capability ($\mathrm{Cpu}(n_i)$), flow rule capacity ($\mathrm{Flow}(n_i)$), and the total bandwidth ($\mathrm{Bw}(l)$) of its adjacent links in the network. According to the topological difference between SN and VN, different node topology metrics are used to quantify the substrate nodes and virtual nodes, respectively. The detailed metrics are defined as follows.

5.1.1. Central Aggregation of Virtual Nodes. This paper uses node proximity clustering degree and node centrality to indicate the topological characteristics of virtual nodes. Node degree is used to represent the node proximity clustering degree, which indicates the number of nodes directly connected to the node. Node centrality represents the centrality of nodes in the global topology. The formulation is defined as follows:

$$C(n_i^v)=\frac{\mathrm{TD}(n_i^v)}{\sum_{j=1}^m d(i,j)}, \tag{23}$$

where $\mathrm{TD}(n_i^v)$ is the number of adjacent nodes of the ranked node. $\sum_{j=1}^m d(i,j)$ is the sum of the shortest distances between the ranked node and all other nodes, which indicate the global topological characteristics of the ranked node.

5.1.2. Closeness Aggregation Degree of Substrate Nodes. In this paper, the closeness degree and the aggregation degree of the substrate node are used to jointly represent the substrate node topology characteristics. The formulation is defined as follows:

$$C(n_i^s)=\frac{\mathrm{TD}(n_i^s)}{\delta+\sum_{n_j\in\psi(n)}d(i,j)}, \tag{24}$$

where $\mathrm{TD}(n_i^s)$ is the number of adjacent nodes of the ranked substrate node, δ is an infinitely small positive number, $\psi(n)$ represents the set of substrate nodes that have already hosted the virtual nodes contained in the VNR, and $d(i,j)$ is the number of hops for the shortest path between two substrate nodes.

It is preferable to embed the virtual node with a higher central aggregation degree first, which can make the virtual node that has a large number of adjacent nodes in the virtual network and close to the center of the network preferentially embedded. On the contrary, embedding virtual nodes onto a substrate node with high closeness aggregation degree can make the virtual nodes in the same virtual network embedded to substrate nodes closer, which can save more bandwidth resources.

5.2. Node-Ranking Methods

5.2.1. Principle of SPA. The core idea of SPA is to regard certainty and uncertainty as a certain uncertain system. It uses identity, discrepancy, and contradistinction components of connection degree to describe the different properties of internal relation between two sets. Therefore, the basis of SPA is set pair, and its key is connection degree.

Suppose the sets $\mathbf{A}=\{x_1,x_2,\ldots,x_n\}$ and $\mathbf{B}=\{y_1,y_2,\ldots,y_n\}$, respectively. The set pair $\mathbf{H}=(\mathbf{A},\mathbf{B})$ is constructed of set $\mathbf{A}$ and relative set $\mathbf{B}$. The connection degree between sets $\mathbf{A}$ and $\mathbf{B}$ can be defined as follows:

$$\mu_{AB}=\frac{s}{n}+\frac{f}{n}I+\frac{p}{n}J=a+bI+cJ, \tag{25}$$

where s is the number of the identical characteristics, p is the number of the contradictory characteristics, and $f=n-s-p$ is the number of the discrepant characteristics. It is marked that $a=s/n$ is the identity degree of sets $\mathbf{A}$ and $\mathbf{B}$, $b=f/n$ is the discrepancy degree, and $c=p/n$ is the contradistinction degree. I is the uncertainty coefficient of

discrepancy, which has different values in [−1, 1] according to different conditions. J is the contradictory coefficient, which has value of −1. The vector (a, b, c) consisting of the identity degree (a), the discrepancy degree (b), and the contradistinction degree (c) is defined as the connection vector of sets **A** and **B**, and it can be denoted as $\mu = (a, b, c)$. The connection vector distance between sets **A** and **B** can be defined as follows:

$$d = \sqrt{(1-a)^2 + b^2 + c^2}. \tag{26}$$

5.2.2. Steps of VNE-NRTD-S Algorithm Based on Modified SPA Method. The SPA method is modified based on the TOPSIS method in this paper, the positive and negative ideal schemes can be obtained by the TOPSIS method, and the relative contact degrees between each scheme and ideal schemes can be obtained by their connection degrees. The steps of VNE-NRTD-S algorithm are detailed as follows.

Step 1. Build a decision-making matrix.

Suppose there are K objects that need to be ranked, which represent different nodes to be embedded. Each node has Q evaluation metrics. The value of the j-th evaluation metric of the i-th node is denoted as X_{ij} $(i = 1, 2, \ldots, K; j = 1, 2, \ldots, Q)$. The decision-making matrix formed by all network nodes and their evaluation metrics is defined as follows:

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1Q} \\ x_{21} & x_{22} & \cdots & x_{2Q} \\ \vdots & \vdots & \ddots & \vdots \\ x_{K1} & x_{K2} & \cdots & x_{KQ} \end{pmatrix}. \tag{27}$$

Step 2. Normalize the decision-making matrix.

Because the dimensions of each metric are different, there may be order of magnitude differences. It is inconvenient to make direct comparisons. The metrics need to be normalized, so as to eliminate the dimensional differences between metrics:

$$y_{ij} = \frac{x_{ij}}{\sum_{i=1}^{K} x_{ij}}. \tag{28}$$

Then, the standard normalization matrix can be defined as follows:

$$\mathbf{Y} = \begin{pmatrix} y_{11} & y_{12} & \cdots & y_{1Q} \\ y_{21} & y_{22} & \cdots & y_{2Q} \\ \vdots & \vdots & \ddots & \vdots \\ y_{K1} & y_{K2} & \cdots & y_{KQ} \end{pmatrix}. \tag{29}$$

Step 3. Calculate the weighted normalized decision-making matrix.

The weight of the j-th evaluation metric can be represented by w_j $(j = 1, 2, \ldots, Q, 0 \le w_j \le 1, \sum_{j=1}^{Q} w_j = 1)$. The weighted normalized decision-making matrix is defined as follows:

$$\mathbf{Z} = \begin{pmatrix} w_1 y_{11} & w_2 y_{12} & \cdots & w_Q y_{1Q} \\ w_1 y_{21} & w_2 y_{22} & \cdots & w_Q y_{2Q} \\ \vdots & \vdots & \ddots & \vdots \\ w_1 y_{K1} & w_2 y_{K2} & \cdots & w_Q y_{KQ} \end{pmatrix}. \tag{30}$$

Step 4. Determine the positive ideal scheme $\mathbf{A}^+$ and negative ideal scheme $\mathbf{A}^-$, and the formulations are defined as follows:

$$\begin{aligned} \mathbf{A}^+ &= \left\{\max\left(z_{i1}, z_{i2}, \ldots, z_{iQ}\right)\right\} \\ &= \left\{z_1^{\max}, z_2^{\max}, \ldots, z_Q^{\max}\right\}, \\ \mathbf{A}^- &= \left\{\min\left(z_{i1}, z_{i2}, \ldots, z_{iQ}\right)\right\} \\ &= \left\{z_1^{\min}, z_2^{\min}, \ldots, z_Q^{\min}\right\}. \end{aligned} \tag{31}$$

The scheme of each node can form a set pair with the positive ideal scheme, denoted as $\mathbf{H}(\mathbf{A}^+, \mathbf{Z}_k)$. Likewise, the scheme of each node can also form a set pair with the negative ideal scheme, denoted as $\mathbf{H}(\mathbf{A}^-, \mathbf{Z}_k)$.

Step 5. Calculate the connection degree of each scheme $\mathbf{Z}_k$ to the positive ideal scheme $\mathbf{A}^+$, and the formulation is defined as follows:

$$\begin{aligned} \mu_k^+ &= a_k^+ + b_k^+ I + c_k^+ J = w_1\mu_{k1}^+ + w_2\mu_{k2}^+ + \cdots + w_Q\mu_{kQ}^+ \\ &= \sum_{t=1}^{Q} w_t \mu_{kt}^+ \\ \mu_{kt}^+ &= a_{kt}^+ + b_{kt}^+ I + c_{kt}^+ J, \quad k = 1, 2, \ldots, K; t = 1, 2, \ldots, Q, \end{aligned} \tag{32}$$

in which when $y_{kt} = z_t^{\min}$, $a_{kt}^+ = b_{kt}^+ = 0$ and $c_{kt}^+ = 1$, and when $y_{kt} \in (z_t^{\min}, z_t^{\max})$, $a_{kt}^+ = y_{kt}/z_t^{\max}$, $b_{kt}^+ = 1 - a_{kt}^+$, and $c_{kt}^+ = 0$.

Step 6. Calculate the connection degree of each scheme $\mathbf{Z}_k$ to the negative ideal scheme $\mathbf{A}^-$, and the formulation is defined as follows:

$$\begin{aligned} \mu_k^- &= a_k^- + b_k^- I + c_k^- J = w_1\mu_{k1}^- + w_2\mu_{k2}^- + \cdots + w_Q\mu_{kQ}^- \\ &= \sum_{t=1}^{Q} w_t \mu_{kt}^- \\ \mu_{kt}^- &= a_{kt}^- + b_{kt}^- I + c_{kt}^- J, \quad k = 1, 2, \ldots, K; t = 1, 2, \ldots, Q, \end{aligned} \tag{33}$$

in which when $y_{kt} = z_t^{\max}$, $a_{kt}^- = b_{kt}^- = 0$ and $c_{kt}^- = 1$, and when $y_{kt} \in [z_t^{\min}, z_t^{\max})$, $a_{kt}^- = z_t^{\min}/y_{kt}$, $b_{kt}^- = 1 - a_{kt}^-$, and $c_{kt}^- = 0$.

Step 7. Calculate the connection vector distance of each scheme to the positive ideal and negative ideal schemes, respectively. The formulations are defined as follows:

$$D_k^+ = \sqrt{\left(1 - a_k^+\right)^2 + \left(b_k^+\right)^2 + \left(c_k^+\right)^2},$$
$$D_k^- = \sqrt{\left(1 - a_k^-\right)^2 + \left(b_k^-\right)^2 + \left(c_k^-\right)^2}. \tag{34}$$

Step 8. Calculate the similarity between each scheme and the ideal one, and the formulation is defined as follows:

$$S_k = \frac{D_k^-}{D_k^- + D_k^+}, \quad 0 \le S_k \le 1. \tag{35}$$

Step 9. Rank the nodes by S_k in decreasing order. The nodes with larger S_k have higher importance in the network.

5.2.3. Steps of VNE-NRTD-G Algorithm. The global bandwidth of the network topology is introduced into the node-ranking process to further improve the performance of the algorithm. The steps of the VNE-NRTD-G algorithm are detailed as follows:

Step 1: for each node in the network, normalize the similarity between each scheme and the ideal scheme:

$$C_k = \frac{S_k}{\sum_{k=1}^{K} S_i}, \quad 0 \le S_k \le 1. \tag{36}$$

Step 2: introduce the global bandwidth of the network topology and calculate the node importance in a recursive manner. The formulation is presented as follows:

$$\mathbf{R}_k = \lambda \cdot C_k + (1 - \lambda) \sum_{m \in \mathrm{Nb}(k)} \frac{\mathrm{Bw}(l_{mk})}{\sum_{n \in \mathrm{Nb}(m)} \mathrm{Bw}(l_{mn})} \cdot \mathbf{R}_m, \tag{37}$$

$$\mathbf{R} = \lambda \cdot \mathbf{C}_\mathbf{R} + (1 - \lambda)\mathbf{M}_\mathbf{R} \cdot \mathbf{R}, \tag{38}$$

$$M(k, m) = \frac{\mathrm{Bw}(l_{mk})}{\sum_{n \in \mathrm{Nb}(m)} \mathrm{Bw}(l_{mn})}. \tag{39}$$

Step 3: rank the nodes by R_k in decreasing order. The nodes with larger R_k have higher importance in the network.

where λ is the damping factor within (0, 1), $\mathrm{Nb}(k)$ indicates the set of the neighbor nodes of noden_k, and $C_k$is the normalized S_k of node n_k in the network.

To calculate the R_k of all nodes, the traffic form of all node importance is expanded and the vector format is defined as follows:

where $\mathbf{R} = (R_1, R_2, \ldots, R_k, \ldots, R_K)^T$. λ is a factor, and $\lambda \in (0, 1)$. $\mathbf{C}_\mathbf{R} = (C_1, C_2, \ldots, C_k, \ldots, C_K)^T$, $\mathbf{M}_\mathbf{R}$ is the transition matrix with dimensions of $K \times K$, each dimension corresponds to the data for the two adjacent points, and $M(k, m)$ is defined as follows:

5.3. Virtual Node Embedding. The VNE-NRTD algorithm proposed in this paper embeds the virtual nodes through greedy way. The detailed procedure is illustrated in Algorithm 1 (VNE-NRTD-S) and Algorithm 2 (VNE-NRTD-G).

5.4. Virtual Link Embedding. If there is a node embedding failure in the VNR, the link embedding is not performed. Otherwise, the *shortest path* algorithm is used to search the shortest path between the nodes. The detailed steps of virtual link embedding algorithm are shown in Algorithm 3.

5.5. Time Complexity of VNE-NRTD. For all virtual nodes in the VNR, the time complexities corresponding to calculating the S_k and R_k of the candidate substrate node are $O(|N^v||N^s|^2)$ and $O(|N^v||N^s|^2 + |N^v||N^s|^2 \log(1/\delta))$ [40], respectively. The time complexity of link embedding is $O(|L^v||N^s|^2)$. Therefore, the time complexity of VNE-NRTD-S is $O(|N^v||N^s|^2 + |L^v||N^s|^2)$, and the time complexity of VNE-NRTD-G is $O(|N^v||N^s|^2 + |N^v||N^s|^2 \log(1/\delta) + L^v||N^s|^2)$.

Overall, compared with other algorithms, the proposed algorithm, either VNE-NRTD-G or VNE-NRTD-S, runs in polynomial time and can be simulated in a continuous time event.

6. Performance Evaluation

In this section, we present the settings of our simulation environment in detail, and then, the performance of our algorithms is compared with three algorithms in the first simulation experiment. Next, the performance of the compared algorithms as a function of different VNR arrival rates is simulated in the second experiment. Finally, we simulate the impact of different VNR duration times on the compared algorithms. Simulation results verify the performance of the VNE-NRTD algorithm.

6.1. Simulation Environment Settings. Similar to the prior work [36], the topologies of the substrate network and virtual networks are generated by the modified Salam network topology random generation algorithm.

Table 2 summarizes the shared SN environments for our simulation. Table 3 summarizes the VN environments for our simulation.

In this paper, our simulation experiments evaluate the performance of five algorithms, which are listed in Table 4. The simulation is performed in the same SN and VNRs environment.

All simulations for different VNE algorithms run on a Windows 7 operating system. The hardware platform is composed of an Intel Core i7-7700 3.6 GHz processor and 8 GB of RAM. The analysis software is MATLAB R2017a. In order to

(1) Calculate the node-ranking value S_k of virtual nodes according to equation (35)
(2) **for** all unembedded virtual nodes in VNR
(3) Select the virtual node n_k with the highest node-ranking value S_k
(4) Rank the virtual nodes using breadth first search (BFS) algorithm and put the ranking results into Virtual_node_list1
(5) **end for**
(6) **for** each unembedded virtual node in Virtual_node_list1
(7) Filter the set of substrate nodes that meet the resource and location constraints of n_k
(8) Calculate the node-ranking value S_k of substrate nodes according to equation (35)
(9) Rank the substrate nodes by S_k in decreasing order
(10) Embed the virtual node onto the substrate node with the highest S_k
(11) **end for**

Algorithm 1: VNE-NRTD-S.

(1) Calculate the node-ranking value R_k of virtual nodes according to equation (37)
(2) **for** all unembedded virtual nodes in VNR
(3) Select the virtual node n_k with the highest node-ranking value R_k
(4) Rank the virtual nodes using BFS algorithm and put the ranking results into Virtual_node_list2
(5) **end for**
(6) **for** each unembedded virtual node in Virtual_node_list2
(7) Filter the set of substrate nodes that meet the resource and location constraints of virtual node
(8) Calculate the node-ranking value R_k of substrate nodes according to equation (37)
(9) Rank the substrate nodes by R_k in decreasing order
(10) Embed the virtual node onto the substrate node with the highest R_k
(11) **end for**

Algorithm 2: VNE-NRTD-G.

(1) Rank the virtual links by the required bandwidth in decreasing order
(2) **for** all unembedded virtual links in VNR
(3) Fetch two corresponding substrate nodes for the virtual link
(4) Remove the substrate links whose bandwidth is less than the required amount of bandwidth
(5) Choose a loop-free substrate path between the corresponding nodes using *shortest path* algorithm
(6) Embed the virtual link onto the choose substrate path
(7) **end for**

Algorithm 3: The link embedding algorithm based on the shortest path algorithm.

Table 2: SN environments.

Network generation approach	Modified Salam algorithm
Number of substrate nodes	100
Number of substrate links	500
Connectivity rate of node	0.5
Computing capability of node	[50, 100], uniform distributed
Flow rule capacity of node	[50, 100], uniform distributed
Link bandwidth	[50, 100], uniform distributed
Location of substrate node	[0, 1000], uniform distributed

reduce the influence of random parameters on the experimental results, we run our simulations for 50,000 time units so that the performance is in a stable state.

6.2. Simulation Results. In this section, we analyze the performance of the proposed algorithm in terms of the VNR acceptance ratio, long-term average R/C ratio, substrate

Table 3: VNR environments.

Network generation approach	Modified Salam algorithm
VNR arrival rate	5 VNRs per 100 time unites
VNR duration time	Mean value is 1000 time unites
Number of VNR nodes	[2, 10], uniform distributed
Node computing capability demand	[10, 30], uniform distributed
Node flow rule capacity demand	[10, 30], uniform distributed
Link bandwidth demand	[10, 30], uniform distributed
Virtual node location constraint	$D(n_i) = 500$

Table 4: Compared algorithms.

Algorithm	Description
VNE-NRTD-G	Virtual node embedding based on global bandwidth metric
VNE-NRTD-S	Virtual node embedding using node multiple metrics based on the modified SPA method
VNE-MC [36]	Virtual node embedding based on resource and topology attributes metrics
VNE-NRM [37]	Virtual node embedding based on resource importance metric
VNE-RCR [37]	Virtual node embedding based on resource balance metric

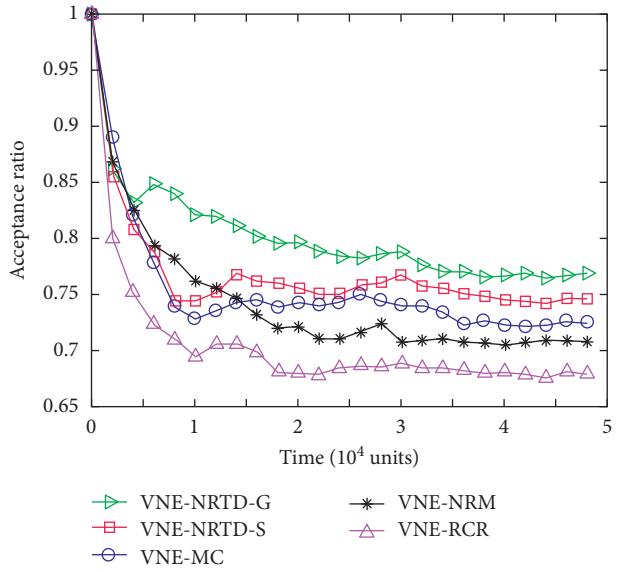

Figure 3: VNR acceptance ratio.

node utilization, and substrate link utilization. The simulation results highlight the superior performance of the VNE-NRTD algorithm.

6.2.1. Simulation Experiment 1. The first experiment aims to evaluate the performance of the VNE-NRTD algorithm compared with the other three state-of-the-art algorithms.

Figure 3 presents the average VNR acceptance ratio as a function of continuous time. Observed from Figure 3, the acceptance ratios of all algorithms are almost equal to 1 at the beginning because there are sufficient SN resources for embedding VNRs. However, as time goes on, the acceptance ratios of all algorithms have begun to decay because there are no infinite resources to embed increasing arrived VNRs. After a while, the acceptance of all algorithms gradually stabilizes because of the arrival and departure of VNRs maintaining a state of dynamic balance. Moreover, the VNE-NRTD-G algorithm outperforms the compared heuristic algorithms. The acceptance ratio of the VNE-NRTD-G algorithm improves 3.4%, 5.9%, 8.2%, and 12.9%, respectively, compared with the algorithms VNE-NRTD-S, VNE-MC, VNE-NRM, and VNE-RCR.

The acceptance ratio of the VNE-RCR algorithm is approximately 0.68. It only considers the resource balance of the nodes, when the node is ranked during the node embedding stage, without considering the resource attributes of the nodes. The VNE-NRM algorithm takes resource attributes into consideration and embeds virtual nodes onto substrate nodes with sufficient resources to improve the acceptance ratio of VNRs. The VNE-MC algorithm uses the BFS algorithm when ranking virtual nodes. The virtual node closest to the embedded node in location and requiring more resources is preferentially embedded to improve the topology association of the substrate nodes. In this way, the link bandwidth cost is reduced, thus improving the acceptance ratio of VNRs.

These three algorithms perform node embedding only combining the nodes' resource attributes and local topology attributes by simple arithmetic operations, while without fully considering the weight of different node metrics in different network environments. As a result, the performance of these algorithms is limited. The VNE-NRTD algorithm proposed in this paper runs as expected because it takes different node metrics to quantify the substrate nodes and virtual nodes, respectively. It also takes the modified SPA method to avoid metric weight imbalance. The iterative operation of global bandwidth further enhances the reliability of node ranking and improves the resource utilization of SN. Compared with the selected heuristic algorithms, the VNE-NRTD performs best.

Figure 4 depicts the long-term average R/C ratio as a function of continuous time. From Figure 4, we can observe that the long-term average R/C ratio of all compared algorithms gradually to be steady with the time goes on. The VNE-RCR and VNE-NRM algorithms have the lowest long-term average R/C ratio. The reason is that the virtual nodes

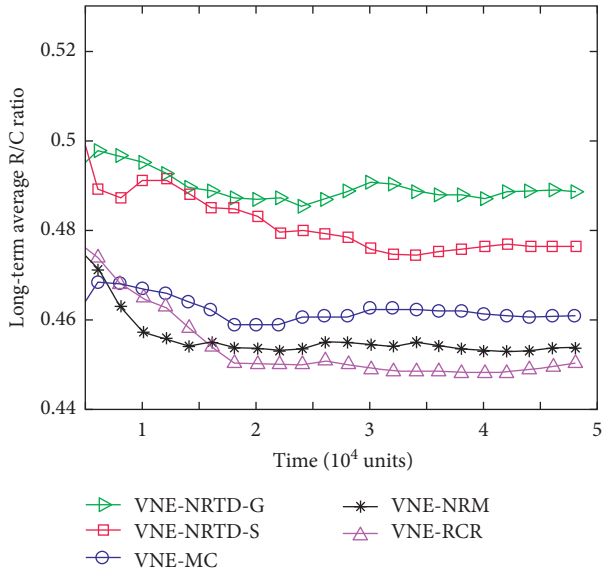

Figure 4: Long-term average R/C ratio.

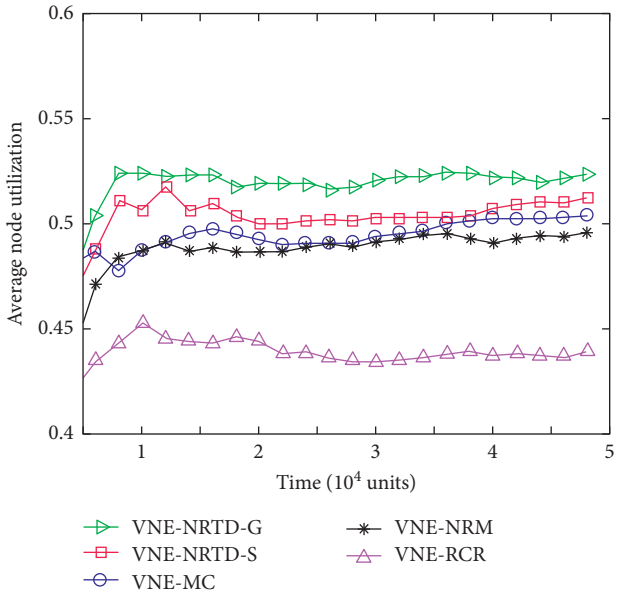

Figure 5: Average node utilization.

are far away after embedding onto the substrate network. Thus, it leads to large amount of unnecessary bandwidth resources consumption. The VNE-MC algorithm considers the topology characteristics of nodes when ranking node to enhance the topology relevance; hence, more substrate link bandwidth resources are saved and the cost of virtual link embedding is reduced. It can be seen from the simulation that the VNE-NRTD algorithm improves the utilization efficiency of substrate network resources and has the highest long-term average R/C ratio, particularly to the VNE-NRTD-G algorithm.

Figures 5 and 6 illustrate the average node and link utilization of all compared algorithms as a function of continuous time, respectively. From Figures 5 and 6, we

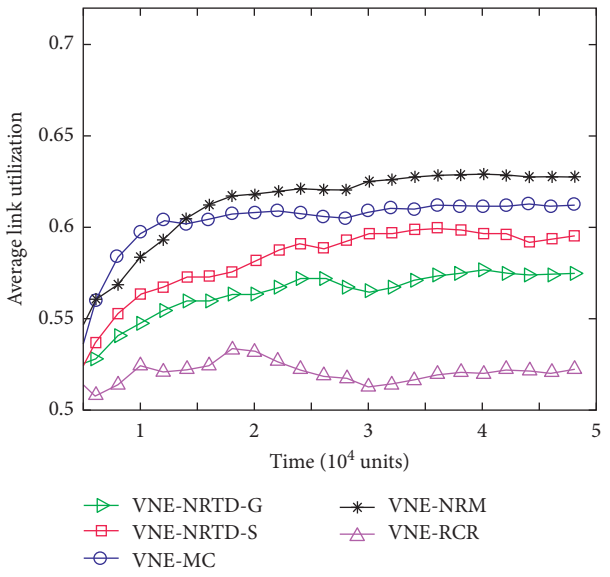

Figure 6: Average link utilization.

can observe that the node and link utilization of all compared algorithms gradually increase with the time. Meanwhile, the VNE-NRTD algorithm has obvious advantages over the compared heuristic algorithms. The reason for the VNE-NRTD algorithm having greater node utilization than other heuristic algorithms is that it can accept more VNRs, and the results in Figure 3 also confirm this inference. However, the link utilization shown in Figure 6 does not have the same result for all algorithms as the previous simulation. The reason for the lower link utilization of the VNE-NRTD algorithm is that it considers the global bandwidth resources when ranking the nodes so that the nodes are closer after embedding, thus avoiding overconsumption of substrate link bandwidth resources. We can also see that the main factor affecting the acceptance ratio of virtual network embedding is that the node resources are finite although the substrate link resources are still sufficient.

6.2.2. Simulation Experiment 2. The second experiment aims to measure the performance of the five compared algorithms as a function of the VNR arrival rate. We compare and analyze the performance of the five algorithms with VNR arrival rate distribution between 1/30 and 1/10.

Figure 7 demonstrates the average VNR acceptance ratio of the five compared algorithms as a function of the VNR arrival rate. From Figure 7, we can observe that the acceptance ratios of all algorithms are decaying with the increasing arrival rate of VNRs. The reason is that the available SN resources are diminishing with the increasing VNRs, which lead to less remaining SN resources that can be used for VN embedding. Moreover, the simulation results in Figure 7 also show that the VNE-NRTD algorithm outperforms the other heuristic algorithms, particularly to the VNE-NRTD-G algorithm, in all simulated VNR arrival rates.

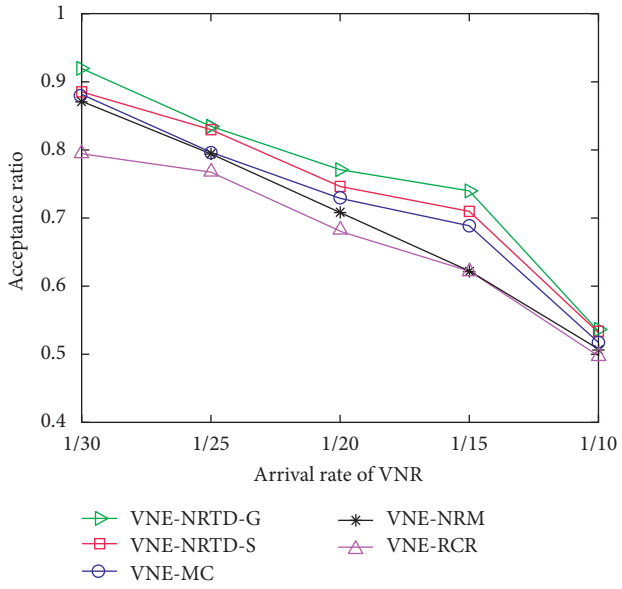

Figure 7: VNR acceptance ratio.

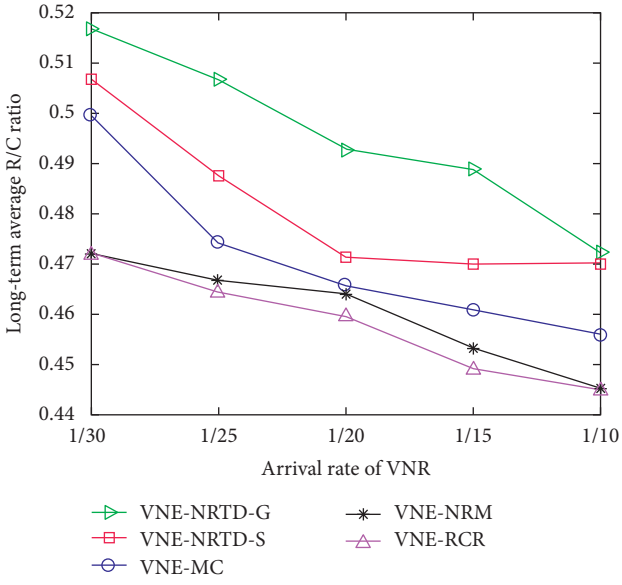

Figure 8: Long-term average R/C ratio.

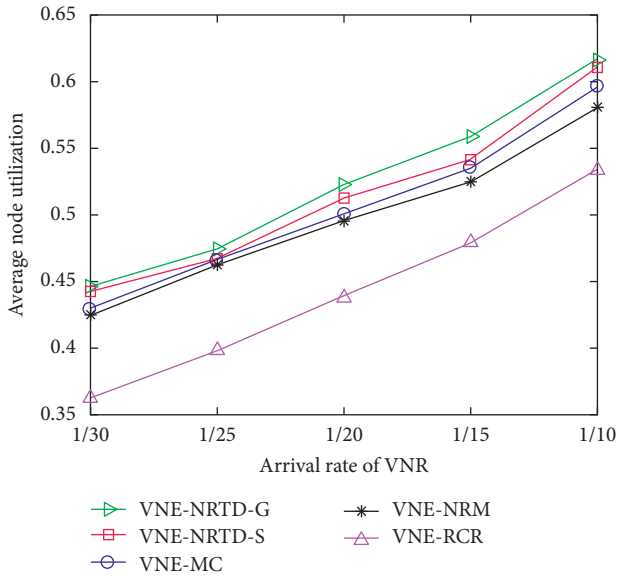

Figure 9: Average node utilization.

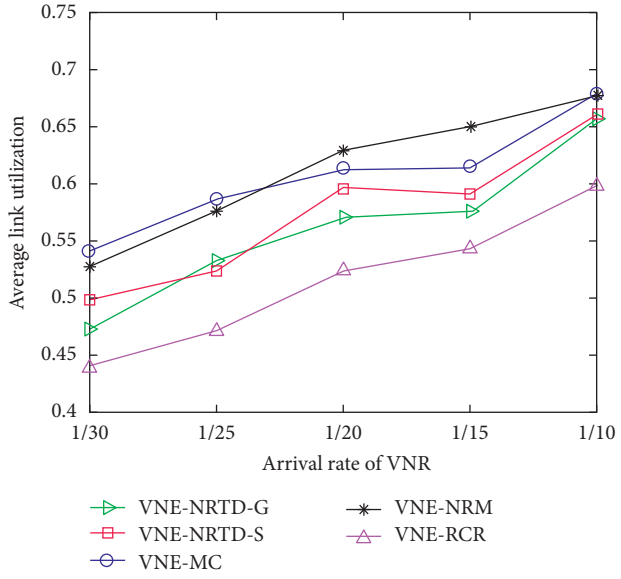

Figure 10: Average link utilization.

Figure 8 shows the long-term average R/C ratio as a function of the VNR arrival rate. From Figure 8, we can observe that the long-term average R/C ratio performance of each algorithm is similar under different VNR arrival rates. The reason is that limited SN resources can only embed a certain number of VNRs. After the VNR is embedded successfully, it will also pay a certain cost while generating revenue. The ratio of revenues to costs will also be stable and balanced. The reason for the VNE-NRTD algorithm having a larger R/C ratio than other algorithms is that the nodes are closer after embedding, which reduce the unnecessary bandwidth consumption of substrate links.

Figures 9 and 10 illustrate the average node and link utilization of all compared algorithms as a function of the VNR arrival rate, respectively. From Figures 9 and 10, we can observe that, with the increase in the virtual network arrival rate, the utilization of the node and link resources of the substrate network has also improved. This is because the greedy strategy used for node and link embedding in the early stage led to resource fragmentation, thus resulted in the inability to fully utilize the remaining resources for virtual network embedding. However, as the arrival rate of VNRs increases, more VNs with fewer resource requirements can be embedded. Therefore, the fragmented SN resources can be fully used for these VNRs for embedding, thereby improving the utilization of substrate network nodes and links resources utilization.

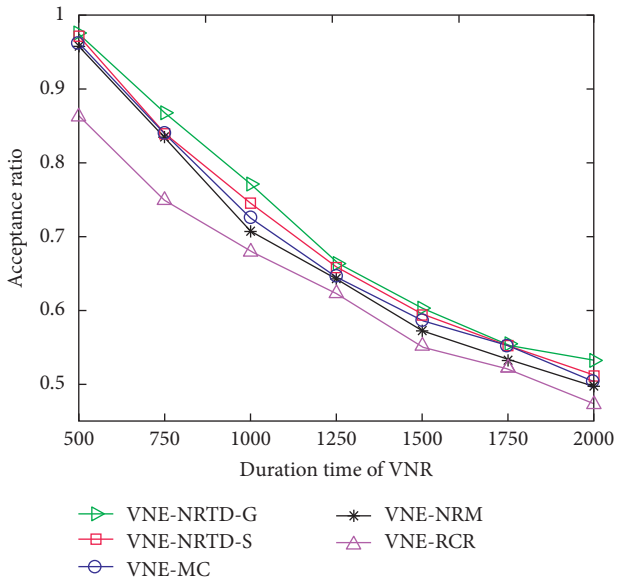

Figure 11: VNR acceptance ratio.

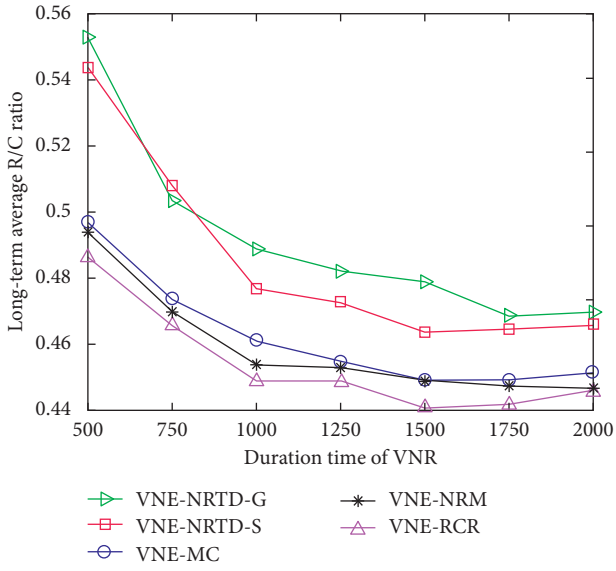

Figure 12: Long-term average R/C ratio.

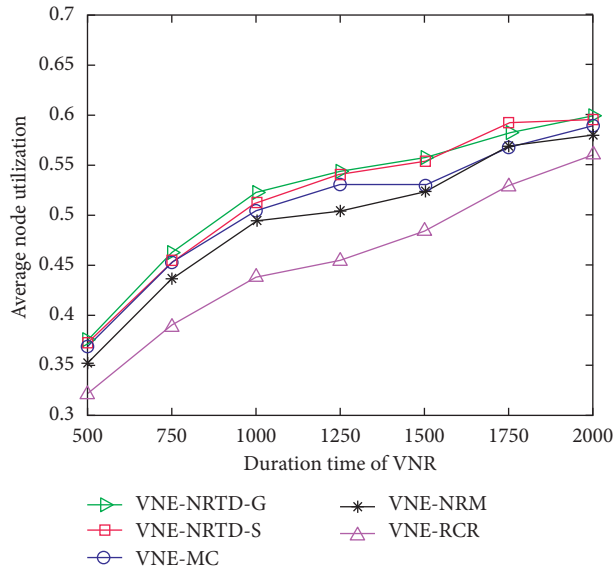

Figure 13: Average node utilization.

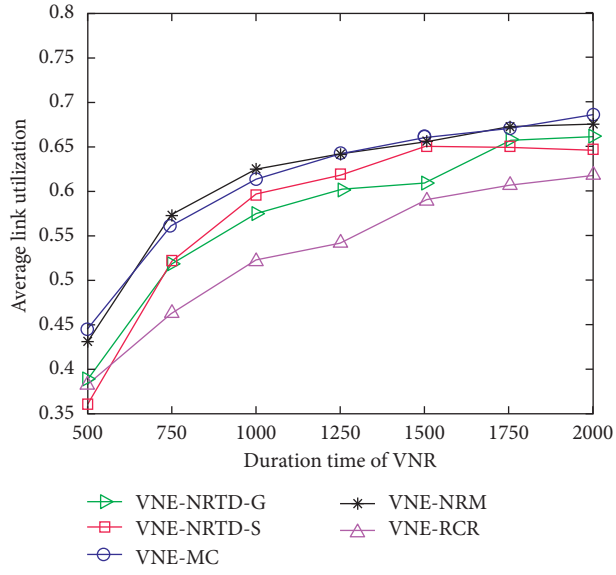

Figure 14: Average link utilization.

6.2.3. Simulation Experiment 3. In order to measure the performance of the five compared algorithms as a function of VNR duration time, we carried out the third experiment and let the duration time of the VNR uniformly distributes between 500 and 2000. Likewise, we evaluate our algorithms compared with the other three algorithms in terms of the average acceptance ratio, the long-term average R/C ratio, and the node and link utilization.

Figure 11 shows the average VNR acceptance ratio as a function of VNR duration time. From Figure 11, we can observe that, as the duration time of VNR increases, the request acceptance ratio of all algorithms gradually decreases. This is because as the duration time of each virtual network request increases, each successfully embedded VNR will occupy SN resources for a long period of time, resulting in insufficient SN resources to embedding more VNRs. The number of VNRs does not change, while the number of successfully embedded VNRs decreases, leading to a lower VNR acceptance ratio. In addition, the VNE-NRTD algorithm can maintain outstanding performance under different VNR durations, especially the VNE-NRTD-G algorithm.

Figure 12 shows the long-term average R/C ratio as a function of VNR duration time. From Figure 12, we can get the same conclusion as in Figure 8. From equation (11), we can observe that the long-term average R/C ratio is also a function of time, while the ratio of revenues to costs will be independent of time. The VNE strategy of each algorithm is fixed, so the ratios of revenues to costs are similar in

different environments. Likewise, the reason for the VNE-NRTD algorithm having larger R/C ratio than other algorithms is that the nodes are closer after embedding, which reduce the unnecessary bandwidth consumption of substrate links.

Figures 13 and 14 illustrate the average node and link utilization of all compared algorithms as a function of VNR duration time, respectively. From Figures 13 and 14, we can observe that, as the duration time of the VNR increases, the node and link resource utilization of the SN increases. The reason is that the VNR with fewer resource requirements is easy to embed, and it will occupy the fragmented resources for a long time, while other VNRs could not use these fragmented resources. Therefore, the node and link resource utilization of the SN is higher when the virtual network request has a long duration time.

7. Conclusion

Virtual network embedding is a key technique to overcome the ossification of the current Internet. To extend the VNE problem to a more realistic approach, we study the virtual network embedding model based on SDN. Then, we proposed a novel node-ranking approach based on topology-differentiating for SDN-based virtual network embedding, labeled as VNE-NRTD, in this paper. According to the topological difference between SN and VN, we used different node metrics to quantify the substrate nodes and virtual nodes, respectively. The importance of nodes is calculated based on the modified SPA method. On this basis, the global bandwidth of the network topology is introduced into the node-ranking process to further improve the efficiency of node embedding. Three kinds of simulation results validate that VNE-NRTD algorithm outperforms the other three typical and latest heuristic algorithms in terms of the VNR acceptance ratio, long-term average R/C ratio, substrate node utilization, and substrate link utilization.

With the continuous development of the future network application, more network parameters need to be considered. The algorithm proposed in this paper can add new parameter indicators flexibly and conveniently, which has better scalability. The proposed algorithm also can be extended to support NFV and other new network scenarios.

In the future work, we will consider more service scenarios to study more challenging virtual network embedding problems, such as node migration and failure recovery, to support multiple services.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

Regarding the publication of this article, the authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. 61873277) and by the Key R&D Project of Shaanxi Province (no. 2020GY-026).

References

[1] R. P. Goldberg, "Survey of virtual machine research," *Computer*, vol. 7, no. 6, pp. 34–45, 1974.

[2] Y. Li and Y. Zhang, "EPVNE: an efficient parallelizable virtual network embedding algorithm," *Wireless Communications and Mobile Computing*, vol. 2019, Article ID 8416592, 10 pages, 2019.

[3] T. Anderson, L. Peterson, S. Shenker, and J. Turner, "Overcoming the Internet impasse through virtualization," *Computer*, vol. 38, no. 4, pp. 34–41, 2005.

[4] P. Zhang, H. Yao, M. Li, and Y. Liu, "Virtual network embedding based on modified genetic algorithm," *Peer-to-Peer Networking and Applications*, vol. 12, no. 2, pp. 481–492, 2019.

[5] A. Fischer and A. Paler, "On the analogy between quantum circuit design automation and virtual network embedding," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pp. 1378–1383, Limassol, Cyprus, April 2019.

[6] T. Chen, J. Liu, Q. Tang, T. Huang, and R. Huo, "Virtual network embedding algorithm for location-based identifier allocation," *IEEE Access*, vol. 7, pp. 31159–31169, 2019.

[7] Q. Hu, Y. Wang, and X. Cao, "Virtual network embedding: an optimal decomposition approach," *in*, in *Proceedings of the 23rd International Conference on Computer Communication and Networks (ICCCN)*, pp. 1–6, Shanghai, China, August 2014.

[8] Z. Li, Z. Lu, S. Deng, and X. Gao, "A self-adaptive virtual network embedding algorithm based on software-defined networks," *IEEE Transactions on Network and Service Management*, vol. 16, no. 1, pp. 362–373, 2019.

[9] H. Cao, L. Yang, Z. Liu, and M. Wu, "Exact solutions of VNE: a survey," *China Communications*, vol. 13, no. 6, pp. 48–62, 2016.

[10] H. Cao, Z. Qu, Y. Xue, and L. Yang, "Efficient virtual network embedding algorithm based on restrictive selection and optimization theory approach," *China Communications*, vol. 14, no. 10, pp. 39–60, 2017.

[11] H. Cao, H. Hu, Z. Qu, and L. Yang, "Heuristic solutions of virtual network embedding: a survey," *China Communications*, vol. 15, no. 3, pp. 186–219, 2018.

[12] H. Cao, H. Zhu, and L. Yang, "Collaborative attributes and resources for single-stage virtual network mapping in network virtualization," *Journal of Communications and Networks*, vol. 99, pp. 1–11, 2019.

[13] Z. Liu, L. Qin, Q. Liu, J. Liu, and Y. Ding, "A hybrid virtual network mapping algorithm based on threshold load," *International Journal of Satellite Communications and Networking*, vol. 37, no. 3, pp. 224–233, 2019.

[14] Z. Wang, Y. Han, T. Lin, H. Tang, and S. Ci, "Virtual network embedding by exploiting topological information," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, pp. 2603–2608, Anaheim, CA, USA, December 2012.

[15] H. Cui, W. Gao, J. Liu, and Y. Liu, "A virtual network embedding algorithm based on virtual topology connection feature," in *Proceedings of the 16th International Symposium on Wireless Personal Multimedia Communications (WPMC)*, pp. 1–5, Atlantic City, NJ, USA, June 2013.

[16] S. Gong, J. Chen, C. Huang, Q. Zhu, and S. Zhao, "Virtual network embedding through security risk awareness and optimization," *KSII Transactions on Internet & Information Systems*, vol. 10, no. 7, 2016.

[17] X. Liu, B. Wang, and Z. Yang, "Virtual network embedding based on topology potential," *Entropy*, vol. 20, no. 12, pp. 1–14, 2018.

[18] L. Shooshtarian and F. Safaei, "A maximally robustness embedding algorithm in virtual data centers with multi-attribute node ranking based on TOPSIS," *The Journal of Supercomputing*, vol. 75, no. 12, pp. 8059–8093, 2019.

[19] C. E. Leiserson, R. L. Rivest, T. H. Cormen, and C. Stein, *Introduction to Algorithms*, MIT Press, Cambridge, MA, USA, 2001.

[20] D. Eppstein, "Finding the *k*-shortest paths," *SIAM Journal on Computing*, vol. 28, no. 2, pp. 652–673, 1998.

[21] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 17–29, 2008.

[22] X. Zheng, J. Tian, X. Xiao, X. Cui, and X. Yu, "A heuristic survivable virtual network mapping algorithm," *Soft Computing*, vol. 23, pp. 1453–1463, 2018.

[23] Y. Zong, Y. Ou, A. Hammad et al., "Location-aware energy efficient virtual network embedding in software-defined optical data center networks," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 10, no. 7, pp. 58–70, 2018.

[24] O. N. Fundation, "Software-defined networking: the new norm for networks," *ONF White Paper*, vol. 2, pp. 2–6, 2012.

[25] Z. Keqin, "Set pair analysis and its prelimiary application," *Exploration of Nature*, vol. 1, 1994.

[26] M. Chowdhury, M. R. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," in *Proceedings of the IEEE INFOCOM*, pp. 783–791, Rio de Janeiro, Brazil, April 2009.

[27] B. Farkiani, B. Bakhshi, and S. Ali MirHassani, "Stochastic virtual network embedding via accelerated Benders decomposition," *Future Generation Computer Systems*, vol. 94, pp. 199–213, 2019.

[28] Z. Yang and Y. Guo, "An exact virtual network embedding algorithm based on integer linear programming for virtual network request with location constraint," *China Communications*, vol. 13, no. 8, pp. 177–183, 2016.

[29] M. Lu, Y. Lian, Y. Chen, and M. Li, "Collaborative dynamic virtual network embedding algorithm based on resource importance measures," *IEEE Access*, vol. 6, pp. 55026–55042, 2018.

[30] I. Fajjari, N. Aitsaadi, G. Pujolle, and H. Zimmermann, "VNE-AC: Virtual network embedding algorithm based on ant colony metaheuristic," in *Proceedings of the IEEE International Conference on Communications (ICC)*, pp. 1–6, Kyoto, Japan, June 2011.

[31] F. Zhu and H. Wang, "A modified ant colony optimization algorithm for virtual network embedding," *Journal of Chemical and Pharmaceutical Research*, vol. 6, no. 7, pp. 327–337, 2014.

[32] I. Pathak and D. P. Vidyarthi, "A model for virtual network embedding across multiple infrastructure providers using genetic algorithm," *Science China Information Sciences*, vol. 60, no. 4, p. 40308, 2017.

[33] J. Liu, T. Song, Y. Hu, and L. Zhuang, "Research on virtual network mapping based on mixed genetic algorithm," *Journal of Chinese Computer Systems*, vol. 37, no. 4, pp. 773–777, 2016.

[34] P. Bosshart, D. Daly, G. Gibb et al., "P4: programming protocol-independent packet processors," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 87–95, 2014.

[35] C. Aguilar-Fuster, M. Zangiabady, J. Zapata-Lara, and J. Rubio-Loyola, "Online virtual network embedding based on virtual links' rate requirements," *IEEE Transactions on Network and Service Management*, vol. 15, no. 4, pp. 1630–1644, 2018.

[36] Z. Zhao, X. Meng, S. Lu, and Y. Su, "Different QoS constraint virtual SDN embedding under multiple controllers," *KSII Transactions on Internet & Information Systems*, vol. 12, no. 9, 2018.

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

[38] A. Blenk, A. Basta, M. Reisslein, and W. Kellerer, "Survey on network virtualization hypervisors for software defined networking," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 655–685, 2015.

[39] S.-q. Gong, J. Chen, Q.-y. Kang, Q.-w. Meng, Q.-c. Zhu, and S.-y. Zhao, "An efficient and coordinated mapping algorithm in virtualized SDN networks," *Frontiers of Information Technology & Electronic Engineering*, vol. 17, no. 7, pp. 701–716, 2016.

[40] H. Cao, L. Yang, and H. Zhu, "Novel node-ranking approach and multiple topology attributes-based embedding algorithm for single-domain virtual network embedding," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 108–120, 2017.