Peer-to-Peer Netw. Appl. (2019) 12:481–492
https://doi.org/10.1007/s12083-017-0609-x

Virtual network embedding based on modified genetic algorithm

Peiying Zhang[1] · **Haipeng Yao**[2] · **Maozhen Li**[3] · **Yunjie Liu**[2]

Received: 17 July 2017 / Accepted: 25 September 2017 / Published online: 13 October 2017

Abstract To fend off the ossification of Internet architecture, virtual network embedding has been propounded as one of the most important techniques to address this issue. Virtual network embedding is a process that consists of two stages including node mapping stage and link mapping stage, the aim of node mapping stage is to map the virtual nodes from virtual network requests (VNRs) onto the substrate nodes meanwhile satisfying the CPU capacity constraints on nodes, the goal of link mapping stage is to map the virtual links from VNRs onto the substrate paths while satisfying the bandwidth resource constraints on links. This paper proposed a virtual network embedding algorithm based on modified genetic algorithm, improved the classical genetic algorithm from three aspects: population initialization strategy, improved mutation operation and improvement operation, took advantage of the selection operation, crossover operation, mutation operation, feasibility checking operation, and utilized the fitness function to choose the best chromosome. Simulation results indicated that our proposed method has significantly increased the acceptance ratio of VNRs and the long-term average revenue of Infrastructures (InPs) compared with other two state-of-the-art algorithms.

Keywords Virtual network embedding · Modified genetic algorithm · Virtual node embedding · Virtual link mapping

This article is part of the Topical Collection: *Special Issue on Software Defined Networking: Trends, Challenges and Prospective Smart Solutions*
Guest Editors: Ahmed E. Kamal, Liangxiu Han, Sohail Jabbar, and Liu Lu

✉ Peiying Zhang
25640521@qq.com

[1] College of Computer & Communication Engineering, China University of Petroleum (East China), Qingdao, China

[2] Beijing University of Posts and Telecommunications, Beijing, China

[3] Electronic and Computer Engineering, School of Engineering and Design, Brunel University, Uxbridge, UK

1 Introduction

With the rapid development of science and technology, many Internet users have the demand of deploying new protocols or installing new services on the existing network architecture. Unfortunately, the current Internet architecture cannot meet the growing demands of the end-users, and the architecture of the current Internet appears rigid phenomenon. Network virtualization technology has been propounded as one of the most important techniques to fend off the ossification. Network virtualization can provision different services for different users through the sliceable management of the existing substrate network resources, and can improve the resource utilization of substrate network and reduce the waste of substrate network resources. Virtual network embedding is the core component of the network virtualization, it mainly concerns on the allocation of substrate resources and it makes the virtual network embedding algorithms the hotspot of the research.

Due to the fact that the problem of virtual network embedding is proved to be a NP-hard problem, domestic and overseas scholars have proposed some of heuristic algorithms to solve this issue or carried out the experiments under the small scale of network topology using the optimization tools. The representatives of these algorithms are

listed in literature [1–4]. The authors of [1] adopted the classical greedy algorithm to map the virtual nodes with larger resource requirements onto the substrate nodes with larger available substrate resources in the node mapping stage, and took advantage of the K-shortest path algorithm or multiple commodity flow (MCF) algorithm to map the virtual links onto the no-loop substrate paths. This algorithm of virtual network embedding is the first time to attempt to adopt the greedy algorithm to address the node mapping problem. The authors of [2] inspired by the Google PageRank algorithm, put forward the topology-aware node ranking algorithm based on the random Markov walk algorithm. Based on this algorithm, the authors proposed two notable virtual network embedding algorithms called as RW-MaxMatch and RW-BFS, respectively. The node ranking model improves the node metric and incorporates the network topology information into the node ranking method, therefore, it can improve the long-term average revenues and increase the ratio of revenue to cost (R/C). Limin Peng in [3] proposed an approach of virtual network embedding based on breadth first search with the purpose of reducing the cost of bandwidth resources for virtual links, this algorithm adopts the adjacent list as the storage structure of substrate network and virtual networks, takes advantage of queue data structure to facilitate the process of virtual network mapping. Because this algorithm uses the strategy of breadth first search to map the virtual nodes onto the adjacent substrate nodes, leads to the average number of hops in the substrate paths which virtual links mapped onto decreased, thus the consumption of bandwidth resources for virtual links is saved. Dong Zhang et al. [4] proposed an algorithm called locality-aware node topological potential ranking (LNTPR) through incorporating the location information of virtual nodes and substrate nodes into the potential function of node ranking, and devised the locality-aware influence choosing node (LICN) algorithm based on a node influence model that considers the mutual influence between a mapped node and its candidate mapping nodes. The combination of LNTPR and LICN improves the integration of node and link embedding.

The authors of [5] applied the particle swarm optimization algorithm to the discrete virtual network embedding algorithm for the first time, they redefined the particle position and operation rules so as to adapt to the discrete virtual network embedding algorithm. This algorithm significantly improved the long-term average revenues and the ratio of revenue to cost (R/C). Liu Jia et al. in [6] presented an approach of virtual network mapping based on mixed genetic algorithm, they utilized the simplex method to estimate the optimal direction, avoid of falling into a local optimum. The drawback of the work [6] is that it utilized the ILOG CPLEX commercial tool to solve the problem in calculating the fitness function of chromosomes. On the one hand, ILOG CPLEX is a commercial software; on the other hand, the computation time is longer when the number of virtual nodes becomes bigger. In this work, we proposed an approach of virtual network embedding based on modified genetic algorithm, this algorithm considers the bandwidth resource utilization of link mapping as the optimization target, and performs the virtual network embedding process by means of selection, crossover, mutation and feasibility checking operations. Extensive simulation results illustrated that our proposed algorithm can significantly increase the long-term average revenue and the acceptance ratio. In addition, the proposed algorithm not only depends on the other commercial software, but also adopted C++ programming language to implement the virtual network embedding algorithm, and reduced the computation time.

The classical genetic algorithm has been applied to many NP-class of problems such as function optimization, combinatorial optimization, and scheduling problems etc. The shortcoming of traditional genetic algorithm is the issue of inefficient and time consuming, we have modified the classical genetic algorithm to address this issue from three aspects: (1) the first aspect is population initialization improvement, the main purpose of our approach is to address the inefficient issue of traditional population initialization. (2) The second aspect is mutation operation strategy, the main purpose of our strategy is to reduce the unnecessary bandwidth resource consumption through choosing mutated substrate nodes from the neighborhood set of the already mapped substrate nodes instead of from all of substrate nodes. (3) The third aspect is a new addition of improvement operation, the main goal of this operation is to further optimize the node mapping solution in order to improve the performance of our algorithm.

We highlight the main contributions and our main ideas as follows:

(1) This work proposed an approach of population initialization that is a trade-off between the time consuming and the particle solution quality. Specifically, we first utilized a dedicated approach to generate a high-quality chromosome solution, and then adopted the mutation operation to generate the other high-quality chromosome solutions, finally analyzed the time complexity of our proposed method.

(2) This work utilized a different strategy of mutation operation with the purpose of reducing the unnecessary bandwidth resource consumption and accelerating the speed of particle position convergence.

(3) This work took advantage of the improvement operation to further optimize the quality of the particle position vector after every iteration operation. The advantage of this step is to improve the node mapping solution and enhance the performance of our algorithm.

The remainder of this paper is organized as follows: In Section 2, we discuss some related works. In Section 3, we introduce the network model and virtual network embedding problems. In Section 4, we describe our proposed algorithm VNE-MGA in detail. Section 5 presents the performance evaluation and analysis. Section 6 concludes the paper.

2 Related works

In this section, we present an overview of the existing literature addressing the virtual network embedding issues. The approaches of virtual network embedding can be divided into two categories according to whether the virtual network requests are known or not in advance. One is the static virtual network embedding approaches in which all of the VN requests are known, and the other is dynamic virtual network embedding approaches in which all the VN requests are generated dynamically.

2.1 Static virtual network embedding approaches

If all of the virtual network requests are known in advance, we call it static virtual network embedding approaches. In the real situations, the virtual network requests arrive randomly, therefore with the purpose of accommodating more virtual network requests for more profits, the SPs contemplate the reconfiguration or remapping the virtual network so as to host the newly arrived VNRs, but the static virtual network embedding approaches do not consider the possibility of remapping the VNRs to obtain more benefits for accepting more VNRs. Even if the virtual network requests are known in advance, the issue of virtual network embedding is also to be a NP-hard problem. The authors of [7, 8] have concentrated on the offline version of virtual network embedding problem. The authors of [7] proposed approaches for specific topologies of backbone-star. They strived for finding the best topology in a family of particular topologies and studied the relative cost-effectiveness of a diverse topologies on a variety of substrate networks, under the broad range of traffic conditions. In the literature [8], the authors suggested a selective VN reconfiguration mechanism that gives the priority to the most critical virtual networks. Extensive simulation results have shown that the proposed algorithms perform better than the other algorithms under a wide range of network conditions.

2.2 Dynamic virtual network embedding approaches

With respect to the static approaches, the virtual network requests do not change during their lifetime, while with respect to the dynamic approaches, the virtual network requests varies with the time. The arrival and departure of the virtual network requests also affect the resource utilization of physical network infrastructure, since several of previously and mapped virtual network requests might cause the substrate resources become scarce. This may result in fragmentation of substrate resources and significantly lower the probability of accommodating newly arrived virtual network requests. To address this issues, different proposals have been suggested in some literatures. Haider et al. [9] investigated the latest techniques for resource allocation of virtual networks, suggested a new objective function for embedding the virtual networks, and fulfilled a systematic discussion of resource allocation problems and identified a variety of significant research challenges. The authors of [10] suggested an approach of VN embedding called VNE-UEPSO which considers two situations that support path splitting and does not support path splitting. In this work, the authors reset the operations and parameters corresponding to the different network conditions, extensive simulation experiments have shown that the proposed algorithms can achieve better convergence and balanced work load and significantly improved the performance of virtual network embedding algorithm in terms of acceptance ratio and long-term average revenue. Wang et al. [11] modified particle swarm optimization to the discrete particle swarm optimization so as to solve the problem of virtual network embedding, they formulated the virtual network embedding issues and abstracted the associated constraints, achieved the optimal solution by means of adapting the characters of virtual network requests.

Recently, several of evolutionary algorithms have been propounded aim at solving many NP-class of problems such as resource scheduling problem, minimum weight triangulation problem, virtual network embedding problem etc. Many researchers employed ant colony optimization algorithm to address the virtual network embedding problem, such as Ant Colony Metaheuristic virtual network embedding algorithm [12], Ant colony optimization based energy efficient virtual network embedding [13], and modified ant colony optimization algorithm [14], and so on. Genetic Algorithm (GA) has been extensively applied to address these complex constraint optimization problems. Virtual network embedding algorithm based on genetic algorithm has been deeply investigated in the literature [15], the authors of [15] proposed two GA based algorithms called CB-GA (based on cost and bandwidth) and RW-GA (based on markov random walk). Extensive simulations indicated that these two GA based methods can produce better performance than PSO-based virtual network embedding algorithms in terms of average infrastructure provider (InP) revenue, acceptance ratio and R/C ratio. The authors of [16] suggested two different representations for each virtual node mapping solution, also offer an analysis of the

influence of different problem representations and in particular the implementation of a uniform crossover for the grouping genetic algorithm that may also be interesting outside of the virtual network mapping domain. A model for virtual network embedding across multiple domains using genetic algorithm is presented in literature [17], simulations illustrated that the proposed model behaves better than other state-of-the-art mapping models.

3 Network model and problem statement

3.1 Substrate network model

The underlying substrate network can be modeled as an undirected weighted graph and denoted by:

$$G_s = \left(N_s, L_s, A_s^N, A_s^L\right) \tag{1}$$

where G_s represents the substrate network, N_s represents the set of substrate nodes in the substrate network, L_s represents the set of substrate links in the substrate network, A_S^N represents the attribute set of substrate nodes in the substrate network, such as the node location, the CPU capacity of substrate nodes, and the storage capacity of substrate nodes, A_S^L represents the attribute set of substrate links in the substrate network, such as the available bandwidth resources, the time-delay of substrate links, or the fluctuation of network traffic.

3.2 Virtual network model

Similarly, each of virtual network request (VNR) also can be modeled as an undirected weighted graph and denoted by:

$$G_v = \left(N_v, L_v, A_v^N, A_v^L\right) \tag{2}$$

where G_v represents the virtual network, N_v represents the collection of virtual nodes in each of virtual network request (VNR), L_v represents the collection of virtual links in the virtual network, A_v^N represents the attribute collection of virtual nodes in the virtual network, such as the required CPU capacity resources or storage capacity resources, A_v^L represents the attribute collection of virtual links in the virtual network, such as the required bandwidth resources.

3.3 Virtual network embedding problem

Virtual network embedding problem is a mapping process which each of virtual network request (VNR) can be mapped onto a shared substrate network, meanwhile satisfying the constraints of CPU computing capacity on nodes and bandwidth resource on links. In most real-world circumstances, the arrival of VNRs is random, therefore, the topologies of virtual network requests are not known in advance. When a virtual network request arrives to the virtual network embedding system, the VNE algorithm has to deal with the VNRs in realtime.

The process of virtual network embedding consists of two stages, including node mapping stage and link mapping stage. In the node mapping stage, we must note that different virtual nodes in the same virtual network request cannot be mapped onto the same substrate node, but different virtual nodes from different virtual network requests can shared the same substrate node. In the link mapping stage, the substrate paths which each of virtual link mapped onto are loop-free. As illustrated in Fig. 1, the substrate network consists of four nodes and five links, these two numbers in the rectangular box are the available CPU capacity resources and the residual CPU capacity resources, respectively. These two numbers across the links are the available bandwidth amount and the residual bandwidth amount, respectively. The virtual network request consists of three virtual nodes and three virtual links. The number next to the node is the required CPU capacity, the number aside the link is the required bandwidth resource.

3.4 Performance evaluation metrics

Virtual network embedding is a multi-objective optimization issue. With respect to the Infrastructure Providers (InPs), they want to accommodate as many virtual network requests as possible using the limited physical resources, efficiently exploit the substrate network resources and make profits from it. These algorithms can be evaluated by means of the long-term average revenue, acceptance ratio and revenue to cost ratio (R/C). We introduce these evaluation metrics as follows.

(1) The long-term average revenue

From the standpoint of InPs, an effective and efficient virtual network embedding algorithm would maximize the revenue of InPs and increase the resource utilization of substrate network in the long run. The metric of long-term average revenue can reflect the effectiveness of embedding algorithms and can be formulated as Eq. 3.

$$R = \lim_{T\to\infty} \frac{\sum_{t=0}^{T}\left(\alpha \times \sum_{n_v\in N_v} CPU(n_v) + \beta \times \sum_{l_v\in L_v} BW(l_v)\right)}{T} \tag{3}$$

where R represents the long-term average revenue of underlying physical network. $CPU(n_v)$ and $BW(l_v)$ are the CPU capacity and bandwidth resource requirements for virtual node n_v and virtual link l_v, respectively. α and β are two weighting factors used for

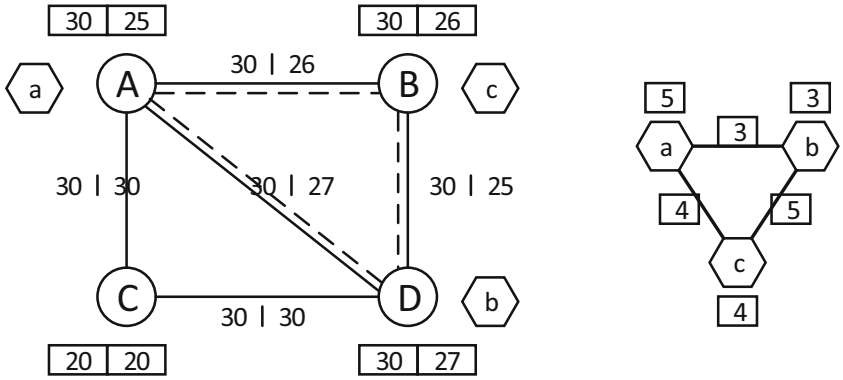

Fig. 1 The illustration of substrate network and a virtual network request

balancing two parts, and they must satisfy the equation $\alpha + \beta = 1.0$. T denotes the running time of virtual network embedding algorithm.

(2) The acceptance ratio of virtual network requests

The acceptance ratio of virtual network requests can reflect the ability to host dynamically arrived virtual network requests. The higher acceptance ratio means this algorithm can provide users with a good user experience, and it can be formulated as Eq. 4.

$$acceptance = \lim_{T\to\infty} \frac{\sum_{t=0}^{T} VN_{accept}}{\sum_{t=0}^{T} VN_{request}} \tag{4}$$

where $acceptance$ represents the acceptance ratio of virtual network (VN) requests, i.e., the ratio of the number of virtual network requests that accepted successfully over the total number of virtual network requests that requested.

(3) The revenue to cost ratio (R/C)

The revenue to cost ratio can reflect the ability to efficiently utilized substrate network resources. The larger R/C value means higher utilization of substrate network resources, and means better performance of the algorithm, and it can be formulated as Eq. 5.

$$R/C = \lim_{T\to\infty} \frac{\sum_{t=0}^{T}\left(\alpha \times \sum_{n_v\in N_v} CPU(n_v) + \beta \times \sum_{l_v\in L_v} BW(l_v)\right)}{\sum_{t=0}^{T}\left(\alpha \times \sum_{n_v\in N_v} CPU(n_v) + \beta \times \sum_{l_v\in L_v} BW(l_v) \times hop(l_v)\right)} \tag{5}$$

where R/C represents the revenue to cost ratio and it is the ratio of revenue that comes from accepting a VN request against the costs that consumed by accommodating a VN request. $hops(l_v)$ denotes the number of hops in substrate path which virtual link l_v mapped onto. The other notations are the same as Eq. 4.

4 Virtual network embedding algorithm based on modified genetic algorithm

4.1 Chromosome encoding

We adopt the genetic algorithm to solve the virtual network embedding problem, the first issue which we faced is the solution to chromosome encoding. Here, we encode each of node mapping solution in the virtual network embedding as a chromosome. With regard to a virtual network request which including n virtual nodes, the chromosome encoding is an N-dimensional vector where each dimension is an integer value which is used to indicate the index of its mapped substrate node for each virtual node. For instance, Fig. 2 represents a node mapping scheme for a virtual network request.

The dimensionality of the chromosome is determined by the number of virtual nodes in each virtual network request, each dimension of the vector specifies the mapping target for each virtual node.

4.2 Population initialization

Definition 1 Population Initialization. After encoding the chromosome is finished, we need to initialize a population as the original solution, the first problem is to determine the number of initial population. The number of initial population is generally obtained from the experiences, and often depends on the number of virtual nodes in the virtual network request lists, the range of population number is generally between 5 and 20.

The main goal of population initialization is to generate a diverse set of solutions with high quality. There are some strategies to initialize the population. The first method is to embed the virtual nodes to the substrate nodes with

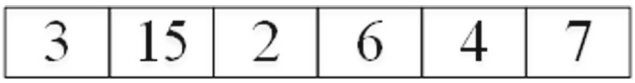

Fig. 2 The example of a chromosome encoding

sufficient network resources randomly. The second method is to exploit the complicated algorithms to generate all of high-quality chromosome solutions. The third method is to utilize a dedicated approach to generate a high-quality chromosome solution, and adopt the mutation operation to generate the other high-quality chromosome solutions. The time complexity of the first method is $O(|N_{pop}| \times |N_v|)$, the complexity of the second method is $O(|N_{pop}| \times (|N_v| \times |N_s|^2 + |L_v| \times |N_s|^3))$, and the complexity of the third method is $O((|N_v| \times |N_s|^2 + |L_v| \times |N_s|^3) + |N_{pop}| \times |N_v| \times |N_s|)$. Therein, $|N_{pop}|$ represents the population number, $|L_v|$ represents the number of virtual links, $|N_v|$ and $|N_s|$ represent the number of virtual nodes and virtual links in each VNR, respectively. The drawback of the first method is that these solutions are lower quality, while the shortcoming of the second method is that it consumes too much computation time. Hence, we adopted the third method to generate the population, and the size of population is set to 10.

4.3 Crossover operation

Definition 2 Crossover Operation. We employ partial mapping crossover to perform the crossover operation, specifically, we classify the parent solutions into some pairwise groups, and each group performs the following steps:

(1) First, we randomly generate two integer values which between one and six, which is denoted by $r1$ and $r2$ (here $r1 = 2$, $r2 = 4$), respectively. And then, we perform the crossover operation on each pair of solutions whose locations between these two integers. The demonstration of crossover operation process is illustrated in Fig. 3.

(2) After finishing the crossover operation, there may be exist some identical serial numbers, we retain the serial numbers if the numbers after crossover operation are different from the existing numbers, otherwise denote the conflict numbers by asterisk (*). The next procedure is to eliminate the conflict serial numbers using the partial mapping method, i.e., utilizing the corresponding relation of the middle segment between $r1$ and $r2$. The final solution after eliminating the conflict serial numbers is illustrated in Fig. 4.

4.4 Mutation operation

Definition 3 Mutation Operation. The mutation operation selects a fraction of already mapped substrate nodes that virtual nodes mapped onto, remap these virtual nodes onto other substrate nodes with sufficient network resources. The fraction of substrate nodes is chosen uniformly in $[0, r]$, where r represents a random real number and here we set it to 0.2. We denote the probability of the mutation operation by P_m and set its value to 0.2. The specific mutation operation procedure is demonstrated in Algorithm 1.

Algorithm 1 The basic mutation operation algorithm

Input:
The set of node mapping solutions, denoted by P.
Output:
The mutated set of node mapping solutions, denoted by PM.

```
for each node mapping solution chr in P do
    generate a random value denoted by rnd uniformly distributed from 0 to 1.
    if rnd < P_m then
        compute the number of nodes which require mutation operation, i.e., N = |N_v| × r.
        i = 0;
        for i < N do
            randomly generate a integer uniformly in [0, |N_v|], and denote it by pos.
            mutate the substrate node in node mapping solution chr at index pos to another subs-trate node that different from these already mapped substrate nodes.
            i = i + 1;
        end for
        put the mutated node mapping solution chr_mutated into the set of PM.
    else
        put the node mapping solution chr into the set of PM.
    end if
end for
```

To illustrate the procedure of mutation operation, we give an example of mutation operation in Fig. 5. During the mutation operation process, the generated integer value denoted by pos has the value of 2, hence, we mutate this substrate node at index $pos = 2$ to another substrate node which different from already mapped substrate nodes in this node mapping solution chr, the newly mapped substrate node should guarantee that its CPU capacity can satisfy the required CPU capacity of virtual node at index 2.

Here we utilize a different mutation strategy to accelerate the speed of particle convergence. In order to reduce the resource utilization of substrate links, we mutated the already mapped substrate nodes onto the other substrate nodes which are closer to the mapped substrate nodes as well as owning sufficient CPU capacities to meet the computing requirements of virtual nodes. The improved mutation operation procedure is demonstrated in Algorithm 1.

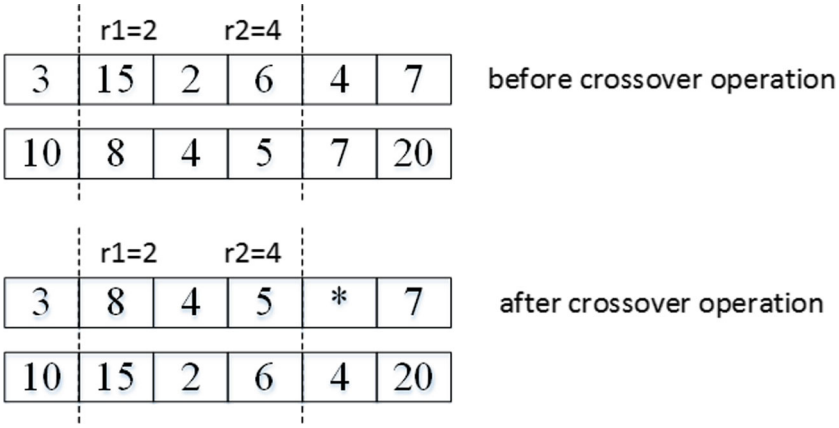

Fig. 3 The illustration of crossover operation

The difference between these two algorithms is in line 8, which imposes some constraints on mutated substrate node with the aim of reducing the unnecessary bandwidth resource waste in the subsequent link mapping process.

Algorithm 2 The improved mutation operation algorithm

Input:
The set of node mapping solutions, denoted by P.
Ouput:
The mutated set of node mapping solutions, denoted by PM.

1: **for** each node mapping solution chr in P **do**
2: generate a random value denoted by rnd uniformly distributed from 0 to 1.
3: **if** $rnd < P_m$ **then**
4: compute the number of nodes which require mutation operation, i.e., $N = |N_v| \times r$.
5: $i = 0$;
6: **for** $i < N$ **do**
7: randomly generate a integer uniformly in $[0, |N_v|]$, and denote it by pos.
8: mutate the substrate node in node mapping solution chr at index pos to another substratenode which are closer to the mapped substrate nodes as well as owning sufficient CPU capacities to meet the computing requirements of virtual nodes.
9: $i = i + 1$;
10: **end for**
11: put the mutated node mapping solution $chr_{mutated}$ into the set of PM.
12: **else**
13: put the node mapping solution chr into the set of PM.
14: **end if**
15: **end for**

4.5 Feasibility checking

Definition 4 Feasibility Checking. We can obtain the newly generated solution after the crossover and mutation operations, but there may be some solutions which violate the constraints of CPU capacity on nodes or violate the constraints of bandwidth resources on links, i.e., there are some solutions which are not feasible solutions. To be able to guarantee that all of the solutions are feasible, we use the feasible checking procedure to eliminate the unfeasible solutions.

Feasibility checking mainly checks the contents of the two aspects: one is to check whether the available CPU capacity of mapped substrate nodes can satisfy the demand of CPU capacity for virtual nodes; the other is to check whether the calculated shortest substrate path can guarantee the required bandwidth resources of virtual links on the situation that the substrate nodes which virtual nodes mapped onto are determined.

4.6 Selection operation

Definition 5 Selecting Operation. If a newly created chromosome falls within the top fraction of the population based on their fitness function values, we chosen these individuals so as to generate the next populations.

4.7 Improvement operation

The nature of genetic algorithm is to inherit important solution properties from the parents to generate high quality solution (i.e. node mapping solution) using crossover, mutation and selection operations. The aim of improvement operation in our work is to improve the performance of virtual network embedding algorithm and accelerate the convergence of our method. The detailed procedures of improvement operation are illustrated in Algorithm 3.

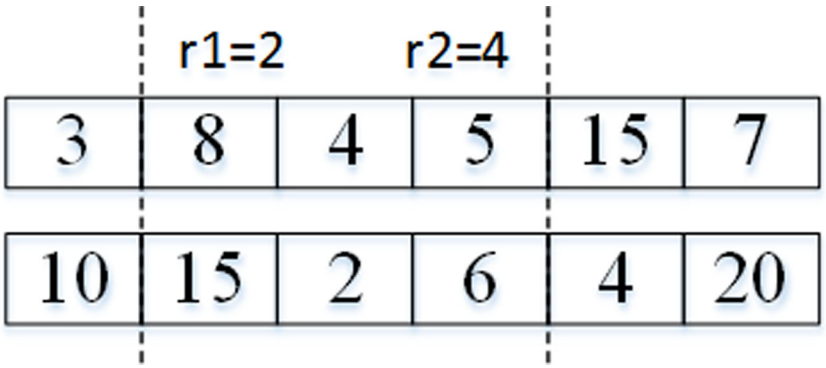

Fig. 4 The illustration of confliction elimination

Algorithm 3 The improvement operation algorithm

Input:

A node mapping solution, denoted by *nodesolution*.

Ouput:

An improved node mapping solution, denoted by *improvednodesolution*.

1: denote the set of substrate nodes by Q, initialize it from the node mapping solution *nodesolution*.
2: choose one substrate node from the set Q, which has the shortest distance summation from the other substrate nodes.
3: map the virtual node corresponding to this selected substrate node onto another substrate node, guaranteeing that the newly mapped substrate node has only one hop distance from the set of already mapped substrate nodes and its CPU capacity resource can satisfy the requirements for the virtual node.
4: denote the modified node mapping solution by *improvednodesolution*.

4.8 Fitness function

Every chromosome can be obtained through encoding the virtual node mapping solutions, therefore, we can obtain the initial virtual node mapping solutions through decoding the chromosomes. Here, we utilize the fitness function to evaluate the quality of each of virtual node mapping solutions in the candidate solution lists, and it can provide a scientific evidence for selecting the chromosomes.

A legal chromosome encoding can determine a virtual node mapping solution in the virtual network requests. With regard to each of the virtual network mapping solutions, the cost of CPU capacity resources for virtual nodes is deterministic, but the cost of bandwidth resources for virtual links is variable since the hops of mapped substrate paths is non-deterministic. Hence, we utilize the cost of bandwidth resources for virtual links as the fitness function and the formulation is as follows:

$$Minimize \quad f(X) = \sum_{l_v \in L_v} BW(l_v) \times hops(l_v) \tag{6}$$

where $f(X)$ denotes the fitness function of each virtual network embedding solution. $BW(l_v)$ represents the bandwidth of virtual link denoted by l_v, $hops(l_v)$ represents the hops of mapped substrate paths corresponding to the virtual link denoted by l_v.

4.9 The proposed algorithm

We denote the fitness function of virtual network embedding algorithm based on modified genetic algorithm called VNE-MGA by $f(X)$, where X represents a legal chromosome encoding, and the chromosome encoding itself can represent a specified virtual node mapping solution for one virtual network mapping instance. In the computation procedure of fitness function $f(X)$, we first check whether the constraints of CPU capacity resources on substrate nodes in the substrate network are satisfied, and then calculate the shortest substrate path corresponding to the virtual links from each virtual network request, third check the feasibility of each virtual network mapping solution. If the solution is infeasible, the fitness function value can be set to infinity; otherwise we utilize the cost of bandwidth resources for

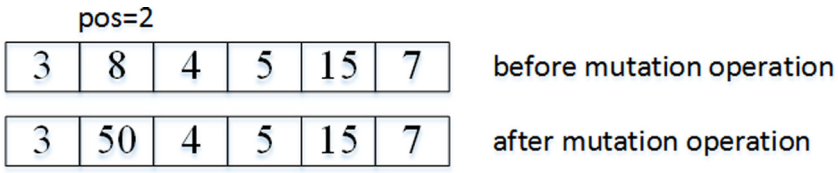

Fig. 5 The illustration of mutation operation

virtual links as the fitness function value. The details of our proposed VEN-MGA algorithm is illustrated in Algorithm 4.

The operations of chromosome encoding include crossover, mutation, feasibility checking, and selection operations. We redefine some of chromosome operations with the purpose of fitting to the virtual network embedding algorithm.

Algorithm 4 virtual network embedding algorithm based on modified genetic algorithm VNE-MGA

Input:
Each virtual network request VNR.
Output:
The solution for each virtual network request VNR.

1: Initialize the chromosome population P, crossover operation probability P_c, mutation operation probability P_m, the computation accuracy δ.
2: $times = 0$;
3: $best\_solution = choose\_best\_solution\_fro\_population(P)$;
4: **while** $times < max\_iteration\_times$ **do**
5: Solution offspring=crossover_operation(P);
6: improved_mutation_operation(offspring);
7: improvement_operation(offspring);
8: **if** feasibility_checking(offspring) **then**
9: insert_population(P,offspring);
10: **end if**
11: $current\_solution = choose\_best\_solution\_from\_population(P)$;
12: **if**$|fitness(current\_solution) - fitness(best\_solution)| < \delta$ **then**
13: break;
14: **else**
15: $best\_solution = current\_solution$;
16: **end if**
17: P=selection_operation(P);
18: $times$++;
19: **end while**
20: return $best\_solution$;

4.10 Time complexity analysis

In our proposed modified genetic algorithm VNE-MGA, the complexity of $choose\_best\_solution\_from\_population(P)$ is $O(|N_{pop}| \times (|N_v| \times |N_s|^2 + |L_v| \times |N_s|^3))$, where N_{pop} represents the population number, N_v represents the number of virtual nodes in each VNR, N_s represents the number of substrate nodes in substrate network, L_v represents the number of virtual links in each VNR. The time complexity of crossover operation is $O(|N_{pop}| \times |N_v|)$, the complexity of improved mutation operation is $O(|N_v| \times |N_s|)$, the complexity of feasibility checking operation is $O(|N_v| \times |N_s|^2 + |L_v| \times |N_s|^3)$, the complexity of fitness computation is $O(|L_v| \times |N_s|^3)$. In addition, Steps 12-16 can guarantee that our proposed algorithm must converge within a certain iteration times.

5 Performance evaluation and analysis

To be able to validate our proposed algorithm VNE-MGA, in this section, we performed the performance evaluation of our proposed algorithm in terms of the aforementioned primary evaluation indexes, and compared with the notable algorithms which denoted by D-ViNE-LB and D-ViNE-SP presented in literature [18].

5.1 Experimental environment setting

Like prior studies did, we adopt the GT-ITM tool to generate the topologies of virtual networks and substrate network. The substrate network contains 100 substrate nodes and about 570 substrate links, the CPU capacity resources and available bandwidth resources are real numbers which uniformly distributed from 50 to 100. The arrival of virtual network requests follows the Poisson process with the average arrival rate of 4 virtual networks per 100 time units. The average survival time of the virtual network requests follows the exponential distribution with the average survival time of 300 time units. The number of virtual nodes for each of VNRs uniformly distributed with U[0,50], and the bandwidth resources of virtual links for each of VNRs are real numbers which uniformly distributed from 0 to 50. The experiment is running for 50000 time units, processes 2500 VNRs, the initial value of population is set to 8, the maximum iteration times $MG = 20$. The crossover probability $P_c = 0.5$ and the mutation probability $P_m = 0.01$.

5.2 Experimental results and analysis

The simulation results are demonstrated by Figs. 6 and 7. In this work, we do not consider the case of path splitting, and evaluate our proposed algorithm and the other algorithm named D-ViNE-SP and D-ViNE-LB in terms of long-term average revenue and acceptance ratio.

Figure 6 illustrates the long-term average revenue of our algorithm VNE-MGA and the other two algorithms D-ViNE-SP and D-ViNE-LB. From the Fig. 6, we can observe that the overall trend of these three algorithms is to decrease first and then increase during the process of simulation experiment. The reason is that the occupied substrate network resources in the first half of simulation experiment are more than those of the second half of simulation

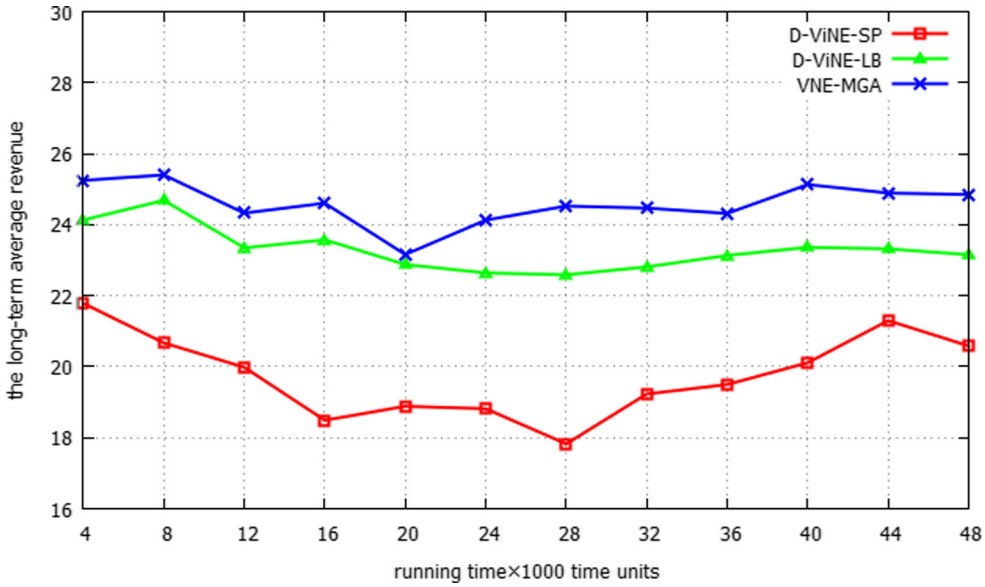

Fig. 6 The long-term average revenue of substrate network

experiment. We can see that our algorithm VNE-MGA is better than the other two algorithms, the reason is that we adopt the improved mutation operation and improvement operation to optimize the node mapping solution, both of these two improvements can reduce the unnecessary bandwidth resource consumption and improve the resource utilization of substrate network, and then increase the long-term average revenue of substrate network.

Figure 7 presents the acceptance ratio of virtual network requests. From the Fig. 7, we can observe that our proposed algorithm VNE-MGA has significantly improved acceptance ratio compared with the other algorithms D-ViNE-SP and D-ViNE-LB. The reason are two folds: on the one hand, to avoid of mapping two adjacent virtual nodes onto two substrate nodes far away from each other, the improved mutation operation can mutate the far substrate nodes into the near substrate nodes with the purpose of reducing the unnecessary bandwidth resource consumption; on the other hand, to further improve the quality of node mapping solution, the improvement operation is added following the improved mutation operation with the aim of improving the solution quality. The saved substrate network resources can satisfy the requirements of new arrival virtual network requests, and then increase the acceptance ratio of VNRs.

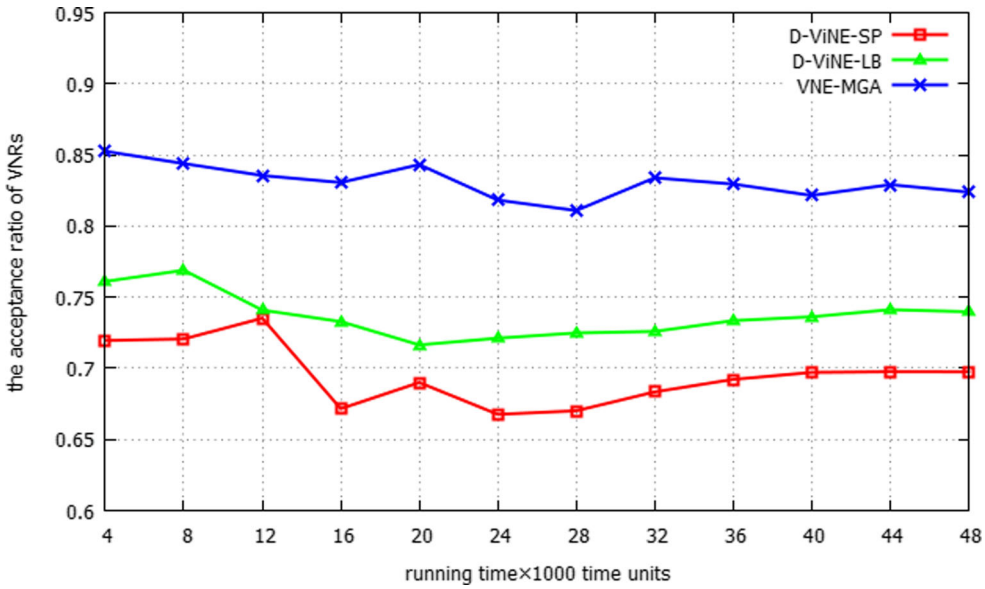

Fig. 7 The acceptance ratio of virtual network requests

6 Conclusion

In this work, we took advantage of bandwidth resource consumption of substrate links as the target function, devised a novel virtual network embedding algorithm based on the modified genetic algorithm VNE-MGA. In our work, we modified the classical genetic algorithm aiming at applying it into the virtual network embedding problem from three aspects. The first aspect is an improved population initialization, the second aspect is the improved mutation operation, and the third aspect is the addition of improvement operation following the mutation operation. The first aspect can reduce the running time of population initialization through analyzing the time complexity of these three methods, the second and third aspects can reduce the bandwidth resource consumption so as to increase the resource utilization of substrate network, and thereby increase the revenue of InPs and acceptance ratio of VNRs. In case of no path splitting, extensive simulation results demonstrated that our proposed algorithm outperforms the other algorithms in terms of the long-term average revenue and the acceptance ratio.

Acknowledgments This work is supported by the Shandong Provincial Natural Science Foundation, China (Grant No. ZR2014FQ018), BUPT-SICE Excellent Graduate Students Innovation Fund, National Natural Science Foundation of China (Grant No. 61471056). The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

Compliance with Ethical Standards

Competing interests The authors declare that they has no competing interests.

References

1. Yu M, Yi Y, Rexford J, Chiang M (2008) Rethinking virtual network embedding: substrate support for path splitting and migration. Acm Sigcomm Computer Communication Review 38(2):17–29
2. Cheng X, Su S, Zhang Z, Wang H, Yang F, Luo Y, Wang J (2011) Virtual network embedding through topology-aware node ranking. Acm Sigcomm Computer Communication Review 41(2):38–47
3. Peng L (2015) Virtual network embedding based on breadth-first search. Sichuan Daxue Xuebao 47(2):117–122
4. Dong Z, Long G (2014) Virtual network embedding through locality-aware topological potential and influence node ranking. Chin J Electron 23(1):61–64
5. Cheng X, Zhang Z, Su S (2011) Virtual network embedding based on particle swarm optimization. ACTA ELECTRONICA SINCA 39(10):2240–2244
6. Liu J, Song T, Hu Y (2016) Research on virtual network mapping based on mixed genetic algorithm. Journal of Chinese Computer Systems 37(4):773–777
7. Lu J, Turner J (2006) Efficient mapping of virtual networks onto a shared substrate, Washington University in St Louis
8. Zhu Y, Ammar M (2007) Algorithms for assigning substrate network resources to virtual network components, in INFOCOM 2006. In: IEEE International Conference on Computer Communications. Proceedings, pp 1–12
9. Haider A, Potter R, Nakao A (2009) Challenges in resource allocation in network virtualization, Itc Specialist Seminar
10. Zhang Z, Cheng X, Su S, Wang Y, Shuang K, Luo Y (2013) A unified enhanced particle swarm optimization based virtual network embedding algorithm. Int J Commun Syst 26(8):1054–1073
11. Wang L, Qu H, Zhao J, Guo Y (2014) Virtual network embedding with discrete particle swarm optimisation. Electron Lett 50(4):285–286
12. Fajjari I, Aitsaadi N, Pujolle G, Zimmermann H (2012) Vne-ac: Virtual network embedding algorithm based on ant colony metaheuristic. In: IEEE international conference on communications, pp 1–6
13. Guan X, Wan X, Choi BY, Song S (2015). In: IEEE international conference on cloud NETWORKING, pp 273–278
14. Zhu F, Wang H (2014) A modified ant colony optimization algorithm for virtual network embedding. J Chem Pharm Res 123(4):68–78
15. Mi X, Chang X, Liu J, Sun L, Xing B (2012) Embedding virtual infrastructure based on genetic algorithm. In: International conference on parallel and distributed computing, applications and technologies, pp 239–244
16. Inf J, Raidl G (2016) A memetic algorithm for the virtual network mapping problem. J Heuristics 22(4):475–505
17. Pathak I, Vidyarthi DP (2017) A model for virtual network embedding across multiple infrastructure providers using genetic algorithm. Sciece China Information Sciences 60(4): 040308
18. Chowdhury M, Rahman M, Boutaba R (2012) Vineyard: Virtual network embedding algorithms with coordinated node and link mapping. IEEE/ACM Trans Networking 20(1):206–219

Peiying Zhang obtained his master degree from China University of Petroleum (East China) in 2006. He is currently a lecturer in the college of computer and communication engineering from China University of Petroleum (East China). He is a PhD candidate in Information and Communication Engineering, from the State Key Laboratory of Networking and Switching Technology in Beijing University of Posts and Telecommunications. His research interests include natural language processing, semantic computing, future internet architecture, network virtualization, and data center network.

Haipeng Yao is currently an associate professor with the School of Information and Communication Engineering, from the State Key Laboratory of Networking and Switching Technology in Beijing University of Posts and Telecommunications. His main research interests include future network architecture, big data for networking, the architecture and key technology of the new generation mobile communication system.

Yunjie Liu received his B.S degree in technical physics from Peking University, Beijing, China, in 1968. He is currently the academician of China Academy of Engineering, the chief of the science and technology committee of China Unicom, and the dean of the school of information and communications in the Beijing University of Posts and Telecommunications. His current research interests include next generation network, network architecture and management.