Application of Improved Multi-Objective Bacterial Foraging Algorithm in Virtual Network Mapping Algorithm

Yihui Qu, Jun Li
School of Electronic and Information Engineering, Lanzhou Jiaotong University, Lanzhou Gansu 730070, China
qyhlove93@sina.cn

Abstract. For the virtual network mapping problem, consideration of many aspects will be more comprehensive. In this paper, the multi-objective bacterial foraging optimization algorithm is improved to solve this problem. First, because the virtual network mapping problem is a discrete problem, the operator of the algorithm is redefined as discrete; secondly, the resource congestion factor and $2-opt$ algorithm are introduced in the chemotaxis operation of the algorithm, and the cross-factor is introduced in the copy operation, so that the found solution is better. At the same time, the node resource load balance degree and the minimum cost are used as the fitness function, so that the found solution solves the problem more. The experimental results show that the algorithm is used to solve the virtual network mapping problem. The algorithm has good validity and stability in both large-scale virtual network requests and standard cases.

1. Introduction
Network virtualization is considered to be the support technology of the future Internet [1-2]. Network virtualization was first proposed by Tom Anderson et al. to change the "rigidity" of the Internet.

The core of network virtualization is to realize the coexistence of multiple heterogeneous virtual networks (VNs) in a physical network infrastructure and share the underlying physical network (SN). Since each virtual network can deploy different protocols, it can help overcome the difficulties faced by the Internet [3].

In network virtualization, the primary entity is the virtual network (VN). Virtual nodes are interconnected by virtual links to form a virtual topology. By virtualizing node and link resources on the underlying physical network, multiple virtual network topologies with widely varying characteristics can be created and co-hosted on the same physical hardware. In addition, the introduction of resource virtualization mechanisms allows network operators to manage and modify networks in a highly flexible and dynamic manner [4-5]. The core problem of resource allocation in network virtualization is Virtual Network Embedding (VNE). Virtual network mapping problems have proven to be NP-hard problems in the literature.

Literature [6] proposed D-vine and R-vine algorithms for cooperative mapping of nodes and links, and solved the mapping problem by relaxing integer constraints. In recent years, some scholars have applied intelligent algorithms to virtual network mapping, such as group intelligent virtual network mapping algorithms based on the idea of particle swarm and artificial bee colony [5,7-10]. In addition, due to the increasingly prominent energy consumption problem in the network, the research on the direction of green energy conservation has appeared in the virtual network mapping problem in recent
years [11-12]. However, no scholar has used the bacterial foraging optimization algorithm to solve this problem.

2. Description of the virtual network mapping problem

2.1. Substrate network (SN)
The underlying physical network topology is represented as a weighted undirected graph \( G_S = (N_S, L_S) \). \( N_S \) represents the set of nodes of the underlying physical network, and \( L_S \) represents the set of links of the underlying physical network. Each underlying physical network node \( n_s \in N_S \) has two weights, one representing the physical location of the physical node \( \text{Loc}(n_s) \) and the other representing the CPU computing power currently available to the physical node \( \text{CPU}(n_s) \); each underlying physical network link \( l_s \in L_S \) has a weight and its weight indicates the bandwidth currently available for this physical link \( \text{BW}(l_s) \).

2.2. Virtual network (VN)
The virtual network topology \( V_N_i \) is also represented as a weighted undirected graph \( G_i = (N_i, L_i) \). \( N_i \) represents the set of nodes of the virtual network \( V_N_i \), and \( L_i \) represents the set of links of the virtual network \( V_N_i \). Each virtual node and each virtual link has a weight indicating the CPU computing power requirement required by the virtual node and the bandwidth requirement required by the virtual link.

2.3. Virtual network requests (VNR)
The \( i \)-th virtual network request \( vnr_i \) is expressed as \( (G_{vi}, t_{wi}, t_{ei}) \). Where \( G_{vi} \) represents the virtual network requesting the mapping, \( t_{wi} \) indicates the time when the virtual request arrived, and \( t_{ei} \) indicates the time when the virtual request leaves. When the virtual network request \( vnr_i \) arrives, the underlying physical network allocates the mapped node CPU resources and link bandwidth resources to the virtual network request \( vnr_i \), but if the node CPU resources and link resources of the underlying physical network are insufficient to be allocated to the virtual network request \( vnr_i \), \( vnr_i \) will be rejected. When the virtual network request \( vnr_i \) leaves, all resources allocated to it are released.

2.4. Virtual network mapping (VNM)
The process of mapping \( V_N_i \) to \( S_N \) can be expressed as \( M : G_i \rightarrow (N_S, P_S) \). Where \( N_S \subseteq N_S \), \( P_S \subseteq \text{Path}_S \), \( \text{Path}_S \) represents the acyclic path of all underlying physical networks. In general, virtual network mapping can be divided into a node mapping phase \( M_N : N_i \rightarrow N_S \) and a link mapping phase \( M_L : L_i \rightarrow P_S \). Usually a virtual network node can only be mapped to one underlying physical network node. An underlying physical network node can only carry one virtual network node requested by the virtual network, but an underlying physical network node can carry multiple virtual network request nodes. Mapping.

3. Solution for virtual network mapping problem

3.1. BFO algorithm and \( \lambda \)-opt algorithm

BFO algorithm
Bacteria Foraging Optimization (BFO) is a biomimetic evolutionary algorithm proposed by Passino in 2002 based on the behavior of E. coli in the human intestine. The algorithm mainly has three steps
of trending, copying and migration. Through the iterative calculation of these three steps, the optimal solution of the problem is found [13].

The behavior of “favoring and avoiding harm” when bacteria are foraging is called chemotaxis. In the process of chemotaxis, bacteria have two basic behaviors: tumble and swim. The movement of the unit in any direction by the unit is defined as a flip. Bacterial flipping is looking for a nutrient gradient. If the bacteria’s value is improved after completing a flip, it will continue to move a few steps in the same direction until the value of the adaptation function is no longer improved, or the threshold value of the preset number of steps is reached, and this process is defined as swimming.

After the chemotaxis operation, the bacteria follow the survival rule of “survival of the fittest, survival of the fittest” in nature. Once the life cycle is over, the critical chemotaxis is reached and the bacteria will multiply. The breeding process of bacteria follows the principle of “survival of the fittest, survival of the fittest” in nature. The cumulative value of the bacterial adaptation values in the chemotaxis process is the standard, and the poor half of the bacteria die, and the better half of the bacteria divide into two sub-bacteria. The sub-bacteria will inherit the biological characteristics of the parent bacteria and have the same position and step size as the parent bacteria. To simplify the calculation, it can be specified that the total number of bacteria remains unchanged during the replication process.

The chemotaxis process ensures the local search ability of bacteria, and the replication process can speed up the search speed of bacteria. However, for complex optimization problems, chemotaxis and replication cannot avoid the local optimal phenomenon of bacteria trapping. BFO introduces the migration process to enhance the overall optimization of the algorithm. After a certain number of copies have been completed, the bacteria will be dissipated to any position in the search space with a certain probability.

3.2. Apply BFO algorithm to solve this problem

Improvements in chemotaxis

Because the virtual network mapping problem is a discrete problem, each bacterium in the algorithm represents a virtual request solution. Each bacterium corresponds to a fitness value, and the algorithm’s related operators need to be redefined.

(1) The location of the bacteria. The position vector of the bacteria $B_{i} = [b_{1, i}, b_{2, i}, \ldots, b_{k, i}, \ldots, b_{N, i}]$ represents the ith possible mapping scheme, N represents the number of virtual network nodes requested by the virtual network, and $b_{k, i}$'s value represents the number of the k-th virtual network node mapped to the underlying physical network node, so its value is taken an A positive integer.

(2) The angle of flipping of the bacteria. Adjust the flip angle of the bacteria to a vector with the same length as the bacterial position vector $A_{i} = [a_{1, i}, a_{2, i}, \ldots, a_{k, i}]$ to adjust the current mapping scheme. $a_{k, i}$ is a binary value. When $a_{k, i} = 1$, the kth virtual network request node needs to reselect
the underlying physical network node. A small number 

\[ u \in \sum_{i=1}^{N} BW_{i}(a_{k}^{i}) + \frac{CPU(a_{k}^{i})}{ALLCPU(a_{k}^{i})} \]

generated while generating each component of the flip angle vector to represent the congestion degree of the underlying physical node, where \( \sum_{i=1}^{N} BW_{i}(a_{k}^{i}) \) represents the sum of the current bandwidths of all the links connected to the physical node \( a_{k}^{i} \), and \( \sum_{i=1}^{N} ALLBW_{i}(a_{k}^{i}) \) represents the total bandwidth of all links connected to the physical node \( a_{k}^{i} \). If the value of \( u \) is the smallest, \( a_{k}^{i} = 1 \), otherwise, \( a_{k}^{i} = 0 \).

After the bacteria adjusted their position, the 2-optimized algorithm was added, and the position of the bacteria was adjusted better after full iteration.

**Improvements in reproduction**

The reproduction operation in the bacterial foraging algorithm combines the single-point crossover operation of the genetic algorithm, that is, the crossover operator selects the mapping scheme of the same virtual request in the two sets of virtual network mapping schemes, and performs the crossover operation. If the overall fitness value is small, it is retained; otherwise, it returns to the original mapping scheme.

**Fitness function**

\[
\begin{align*}
\min \left\{ \begin{array}{l}
    f_{1} = \sum_{l_{uv} \in L_{v}} LEN(l_{uv}) \cdot BW(l_{uv}) \\
    f_{2} = \frac{\sum_{n_{v} \in \mathcal{N}_{v}} SPCPU(n_{v}) - SPCPU(n_{v})}{\sum_{n_{v} \in \mathcal{N}_{v}} ALLCPU(n_{v})}
\end{array} \right.
\end{align*}
\]

In this paper, \( f_{1} \) and \( f_{2} \) are used as fitness functions. Where \( f_{1} \) represents the bandwidth cost of the virtual network mapping scheme; \( f_{2} \) represents the load balancing degree of the remaining underlying physical network nodes, \( SPCPU(n_{v}) \) represents the resources used by the current physical node, \( SPCPU(n_{v}) \) represents the average usage of the physical node resources, and \( ALLCPU(n_{v}) \) represents the total resources of the physical node.

**3.3 Main algorithm flow**

1. Initialization parameters, scale \( N \) of the bacterial population, number of trending operations \( Nc \), number of copy operations \( Nre \), maximum number of attempts \( Ns \) in the trending operation. Initialize the underlying physical network and divide it into four major areas.

2. The virtual network request arrives, the bacterial group is grouped, and the virtual network node is randomly mapped to the underlying physical network node in an area according to the virtual network request allocated by each group of bacteria.

3. Calculate the shortest path between each virtual network link mapped to the pair of underlying physical network nodes according to the k-shortest path algorithm. If the mapping is successful, allocate the node resource and the link resource to the virtual network request, and go to step (4); if the mapping fails and does not exceed the maximum number of attempts, go to step (2), if the maximum number of attempts is reached, then Virtual network request mapping failed.

4. Calculate the fitness values \( f_{1} \) and \( f_{2} \) of the bacteria under this scheme.

5. Carry out the chemotaxis operation, compare the fitness value of the newly formed bacteria with the original bacteria, and replace the original bacteria if the new individual is superior, otherwise no change will be made. If the maximum number of times of the trend operation \( Nc \) is not reached, the process proceeds to step (5); if it is reached, the process proceeds to step (6).

6. Perform a reproduction operation. If the maximum number of reproduction operations \( Nre \) has not been reached, go to step (5); if yes, go to step (7).

7. Output a virtual network mapping scheme according to user requirements.
4. Experimental results and analysis

4.1. Experiment setup
This paper sets up three underlying physical network topologies, one with 100 physical nodes and about 500 physical links, one with 75 physical nodes and about 375 physical links, and one with 50 physical nodes and about 250 physical chains. The three underlying physical network topologies have their node resources and link resources obeying the uniform distribution of [50,125], the uniform distribution of [50,100] and the uniform distribution of [50,75]. We assume that the arrival of virtual network requests per 100 time units is subject to a Poisson distribution with an average of 5 and 10 on each of the three underlying physical networks. The average lifetime of these virtual network requests is 500 time units. The number of virtual network request nodes is uniformly distributed according to [2, 10], and the connection probability of each pair of virtual network request nodes is 0.5, and the virtual network request node CPU demand resources and link demand bandwidths are subject to random distribution [0, 30]. All of the above network topologies and their positional coordinate information are generated by the GT-ITM tool [15], and each simulation experiment runs about 10,000 time units.

When the arrival of the virtual network request per 100 time units is subject to a Poisson distribution with an average of 10, the algorithm population is set to 60; and when the acceptance of the virtual network request per 100 time units is accepted, the average is 5. When the Poisson distribution is set, the algorithm population is set to 30. The number of bacterial chemotaxis was 30, the number of bacterial replications was 20, and the maximum number of adjustments in the same direction was 10 times.

4.2. Evaluation index
(1) Income to expense ratio.

After the mapping is successful, all virtual network request links correspond to the bandwidth of the underlying physical network link and the sum of the CPU resources of the underlying physical network node corresponding to all virtual network request nodes, namely:

\[ \text{cost} = \sum_{l_{uv} \in L_v} \text{LEN}(l_{uv}) \cdot \text{BW}(l_{uv}) + \sum_{n_v \in N_v} \text{CPU}(n_v) \]

Where u and v are the two ends of the virtual link, \( \text{LEN}(l_{uv}) \) is the mapped physical link length of the mapping, \( \text{BW}(l_{uv}) \) is the virtual network requesting the link request bandwidth, and \( \text{CPU}(n_v) \) is the CPU resource required by the virtual network requesting node \( n_v \).

Revenue refers to the resource requirements of the virtual request. The virtual network requests all the link demand bandwidth resources and all the node requirements CPU resources:

\[ \text{revenue} = \sum_{l_v \in L_v} \text{BW}(l_v) + \sum_{n_v \in N_v} \text{CPU}(n_v) \]

Where \( l_v \) is the virtual link and \( \text{BW}(l_v) \) is the bandwidth requirement of \( l_v \). Then the income cost ratio is expressed as:

\[ \text{ratio} = \frac{\text{revenue}}{\text{cost}} \]

(2) The balance of the underlying physical network nodes. The load balancing of the remaining underlying physical network nodes.

\[ \text{LB} = \frac{\sum_{n_s \in N_s} |\text{SPCPU}(n_s) - \overline{\text{SPCPU}}(n_s)|}{\sum_{n_s \in N_s} \text{ALLCPU}(n_s)} \]

(3) Request acceptance rate. The case where the virtual network requests the success or failure of the mapping.

\[ \text{RP} = \frac{\sum_{t=0}^{T} V_s}{\sum_{t=0}^{T} V} \]
Where $\sum_{t=0}^{T} V_s$ represents the number of successful virtual network request mappings from time $t = 0$ to time, and $\sum_{t=0}^{T} V$ represents the total number of virtual network requests arriving from time $t = 0$ to time $T$.

(4) Average income.

$$AVGR = \frac{\text{revenue}}{T}$$

4.3. Performance analysis

Fig. 1 Comparison of three underlying physical network request acceptance rates

Figure 1 shows a comparison of the Poisson distribution request acceptance rates for the average of 5 and 10 arrivals for virtual network requests per 100 time units on each of the three underlying physical networks. As can be seen from Figure 1, whether the number of virtual network requests with an average of 5 or the number of virtual network requests with an average of 10 is accepted, the request acceptance rate is always better than the other two on the underlying physical network with a physical node of 100. The underlying physical network. This indicates that the virtual request acceptance rate is related to the size of the underlying physical network, and the number of virtual requests is unchanged. When the underlying physical network scale is expanded, the virtual request acceptance rate will also increase. At the same time, we can see from the two figures that when the virtual network request comes in a large number, the virtual request acceptance rate of all the underlying physical networks is greatly reduced. However, after some virtual network requests left and released some resources, the acceptance rate increased and stabilized.

Fig. 2 Comparison of the average revenue-to-cost ratio of the three underlying physical networks

(a) The virtual request arrives at a Poisson distribution with a mean of 5.

(b) The virtual request arrives at a Poisson distribution with a mean of 10.
Figure 2 shows a comparison of the Poisson distribution average revenue cost ratios for the average of 5 and 10 arrivals per virtual network request on each of the three underlying physical networks. As can be seen from Figure 2, whether it is accepting the number of virtual network requests with an average of 5 or the number of virtual network requests with an average of 10, the average revenue cost ratio on the underlying physical network with a physical node of 50 is always better than the other two. The underlying physical network is due to the fact that the underlying physical topology with 50 physical nodes is more concentrated than the underlying physical topology nodes with 75 and 100 physical nodes.

![Fig.3 Comparison of average load balance of three underlying physical networks](image)

(a) The virtual request arrives at a Poisson distribution with a mean of 5.
(b) The virtual request arrives at a Poisson distribution with a mean of 10.

Figure 3 shows a comparison of the average Poisson distribution load balance of the average of 5 and 10 arrivals per virtual network request on each of the three underlying physical networks. It can be seen from Figure 3 that the average load balance is high on the underlying physical network with a physical node of 50, especially when the arrival of the incoming virtual network request per 100 time units obeys a Poisson distribution with an average of 10. All nodes on this underlying physical network are almost always full.

![Fig.4 Comparison of average earnings of three underlying physical networks](image)

(a) The virtual request arrives at a Poisson distribution with a mean of 5.
(b) The virtual request arrives at a Poisson distribution with a mean of 10.

Figure 4 shows a comparison of the average revenue of Poisson distributions with average arrivals of 5 and 10 for each of the three underlying physical networks that accept virtual network requests per 100 time units. As can be seen from Figure 4, when the virtual network requests a large number of arrivals, the average revenue of all the underlying physical networks is greatly reduced, but after some
virtual network requests leave and release some resources, the acceptance rate rises and tends to smooth.

5. Conclusion
In this paper, a multi-objective bacterial foraging optimization algorithm is proposed to solve the virtual network mapping problem. The algorithm introduces the $2-\text{opt}$ algorithm to adjust the cost of the algorithm in the chemotaxis operation, and introduces the crossover operator of the genetic algorithm in the copy operation. Adjust the cost of the algorithm and the balance of node resources. After experimental comparison, this algorithm has certain effectiveness, and can maintain the acceptance rate as much as possible when a large number of virtual network requests arrive, while optimizing cost and node resource balance.

References
[1] Chowdhury N M M K, Boutaba R. Network virtualization: state of the art and research challenges[J]. Communications Magazine IEEE, 2009, 47(7):20-26.
[2] Cheng Xiang, Zhang Zhongbao, Su Sen et al. A review of virtual network mapping problems[J]. Journal of Communications, 2011, 32(10):143-151 (in Chinese).
[3] Li Xiaoling, Wang Huaimin, Ding Bo et al. Research and Development of Virtual Network Mapping Problem[J]. Journal of Software, 2012(11):3009-3028 (in Chinese).
[4] Fischer A, Botero J F, Beck M T, et al. Virtual Network Embedding: A Survey[J]. IEEE Communications Surveys & Tutorials, 2013, 15(4):1888-1906.
[5] Shahin A A. Memetic Multi-Objective Particle Swarm Optimization-Based Energy-Aware Virtual Network Embedding[J]. International Journal of Advanced Computer Science & Applications, 2015, 6(4).
[6] Mosharaf N M, Chowdhury K, Rahman M R, et al. Virtual network embedding with coordinated node and link mapping[C]// Infocom. DBLP, 2009.
[7] CHENG Xiang, ZHANG Zhong-bao, SU Sen et al. Virtual Network Mapping Algorithm Based on Particle Swarm Optimization[J]. Chinese Journal of Electronics, 2011, 39(10): 2240-2244 (in Chinese).
[8] Hu Ying, Zhuang Lei. Application of Improved Particle Swarm Optimization Algorithm in Virtual Network Mapping[J]. Computer Engineering and Science, 2014, 36(11): 2169-2173 (in Chinese).
[9] Chen L W. A Virtual Network Mapping Method Based on Cloud Modified Particle Swarm Optimization[J]. Journal of Inner Mongolia Normal University, 2016.
[10] Jia Wei, Xia Jingbo. Research on Cross-domain Virtual Network Mapping Problem[J]. Journal of Electronics & Information Technology, 2016, 38(3): 728-734 (in Chinese).
[11] Hu Ying, Zhuang Lei, Chen Hongceng et al. Bayesian virtual network mapping of time and energy perception[J]. Journal of Communications, 2016, 37(6): 106-118 (in Chinese).
[12] Melo M, Sargento S, Killat U, et al. Optimal virtual network embedding: Energy aware formulation[J]. Computer Networks the International Journal of Computer & Telecommunications Networking, 2015, 91(C): 184-195.
[13] LI Jun, DANG Jian-wu, BU Feng. Research and Improvement of Bacterial Foraging Optimization Algorithm[J]. Journal of Computer Simulation, 2013, 30(4): 344-347 (in Chinese).
[14] Wang Yongzhen, Chen Yan, Li Tao. Discrete bacterial foraging algorithm for solving TSP[J]. Journal of Computer Applications, 2014, 31(12): 3642-3645 (in Chinese).
[15] Zhang Chunhuan, Lei Lei, Ji Yuefeng. A New Network Simulation Topology Generator[J]. Journal of System Simulation, 2006, 18(11): 3115-3117 (in Chinese).