Energy-Driven Virtual Network Embedding Algorithm Based on Enhanced Bacterial Foraging Optimization

ZEXI XU*, LEI ZHUANG, SHUAIKUI TIAN*, MENGYANG HE, SIJIN YANG, YU SONG, AND LING MA
School of Information and Engineering, Zhengzhou University, Zhengzhou 450001, China
Corresponding author: Lei Zhuang (ielzhuang@zzu.edu.cn)

This work was supported in part by the Key Program of the National Natural Science Foundation of China under Grant U1604262, in part by the Key Scientific Research Program of Higher Education of Henan under Grant 19A520003, and in part by the Key Research and Development and Promotion Project in Science and Technology of Henan under Grant 182102210189.

ABSTRACT One of the core challenges facing network virtualization is how to manage the underlying resources to host more virtual networks (VNs) with less energy. With the goal of reducing the energy consumption of virtual network embedding (VNE) and ensuring the VNE solution validity, this paper presents a new approach based on the bacterial foraging optimization (BFO) algorithm devoted to using the underlying resources more effectively and reasonably. In particular, by mimicking the information interaction between bacteria in the process of foraging, we devise a topology-aware mapping strategy, known as SBR_E, to build the candidate set of the VNE solutions. Then, we leverage SBR_E to design the energy manager of the optimization process and develop an energy-driven VNE algorithm based on BFO identified as EBFO to search for the optimal solution. Experimental results show that our method is superior to the existing algorithms in terms of reducing energy consumption and improving the VN request acceptance rate and revenue-cost ratio.

INDEX TERMS Network virtualization, energy-driven, bacteria foraging optimization, energy-efficient.

I. INTRODUCTION
Network virtualization is a promising technology that helps overcome shortcomings, such as network ossification of the current Internet architecture, and can realize efficient and dynamic network service deployment and management. Network virtualization has played an important role in the development of the 5G network. This technology separates the classic ISP (Internet Service Provider) into two new actors: the Service Provider (SP) and the Infrastructure Provider (InP) [1]. Based on this, the virtual network embedding (VNE) problem can be presented as an SP “renting” some resources of the substrate network (SN) from an InP and providing services to the end user. It can be further described as embedding virtual nodes and links in a sequence of VNs to the shared SN, and this has been extensively studied [2]. However, since the resources of SN are limited, to increase revenues, InPs must reasonably and effectively utilize the substrate resources to process each online virtual network request (VNR) on demand. Therefore, building virtual networks (VNs) and allocating the underlying resources for VNRs of different topologies, resource requirements, and lifetimes is an issue of great concern for InPs and is the current research hotspot [6].

VNE has proven to be an NP-hard problem. Finding an exact optimal solution for this type of problem usually incurs exponential running time, large complexity, and requires many restrictions on the problem space, not scalable for large scenarios [10]. Therefore, currently, inspired by the wisdom of nature and based on the functions, characteristics, and mechanisms of organisms, some scholars employ a class of models and algorithms with adaptive, self-organizing, and self-learning abilities to solve various complex problems that are difficult to solve using traditional computing methods. For example, imitating the foraging behavior of natural organisms led to the creation of the particle swarm optimization algorithm [15], artificial bee colony algorithm [16], and ant colony algorithm [17]. These methods tend to find feasible
solutions that approximate the best options and that have been widely applied in solving the VNE problems (see our previous work [18], [19] and the related work [20]–[24]). However, the swarm intelligence algorithms in the previous studies did not completely consider the communication between the individuals in the period of individual foraging. As such, these algorithms have the disadvantages of easily falling into a local optimal solution and having a slower convergence speed. Moreover, in the VNE problem, to use the underlying resources reasonably and effectively, a strategy must be developed to monitor the energy consumption throughout the whole process to realize the goal of reducing the energy consumption while ensuring a high request acceptance rate and revenue-cost ratio. In previous studies on both exact and approximate algorithms [25]–[31], most energy-saving algorithms solve the VNE problem by focusing on the network topology. Although some related studies [25]–[27] are topology-aware, they focus on the availability of substrate resources or accessibility of resources, not on their connection or the structure of the network topology. The nodes selected in this manner may have a scattered location, which will result in the use of more substrate resources. With the substrate resources powered up and links segmented, the number of active nodes and links rises, the utilization rate of resources decreases, and the mapping success rate decreases, as well. Most researchers set the energy manager only in the mapping phase [28]–[31]. In fact, in most two-stage VNE algorithms, they set the manager in the node mapping phase only, often overlooking the optimization phase. As shown in Fig. 1- (a) (where E-A represents energy-aware), in the iterative optimization phase, researchers do not anticipate resource constraints and set every update to be energy-driven to monitor energy consumption. This approach can result in the following phenomena – with iterative updates, the topology of the VN’s physical host network, which was originally proposed under its energy manager, would be constantly changing. In this situation, it is difficult to guarantee both at the same time – the two contradictory metrics of energy efficiency and the high mapping success rate. Therefore, the most energy-efficient solution can only be proposed at the expense of the solution’s quality because as the node position and the situation of substrate network change all the time, in the pursuit of an energy-saving solution, the connectivity between nodes will be difficult to guarantee at the same time. Therefore, a new VNE method with an integrated structure is required to complete the whole embedding process with greater energy awareness (Fig. 1- (b)) to find the optimal solution for the VNE problem.

Compared with other traditional optimization algorithms, bacterial foraging optimization (BFO) [32] has an integrated structure and a low coupling between the modules. As a clear and simple meta-heuristic algorithm, BFO has been proven to be an effective method for finding approximate optimal solutions for NP hard problems [33], [34]. BFO not only has the computing ability of swarm intelligence but also has the ability to break away from the local optimum. Therefore, this paper aims to reduce the energy consumption of VNE while ensuring the validity of the solution. By mimicking the information interaction scenario of BFO, we first designed a topology-aware mapping strategy identified as SBR_E to build the candidate set of the VNE solutions. Then, we leverage SBR_E to design an energy-driven VNE algorithm based on BFO identified as EBFO to approximate the optimal solution.

The main contributions of this paper are as follows:
1. A novel energy-driven VNE method based on the enhanced BFO is proposed, devoted to making the underlying resources used effectively and reasonably.
2. By mimicking the information interaction between bacteria in the process of foraging, we designed a topology-aware mapping strategy SBR_E. It selects high priority underlying resources in the order of socialite → backbone network → remaining links to host the virtual network requests and build the candidate set of the VNE solutions.
3. To find the optimal solution, we designed a VNE algorithm identified as EBFO. It leverages SBR_E to design the energy manager, which is used to guide this optimization process through each step of energy-driven execution to monitor the overall energy consumption. Moreover, to increase the convergence speed, we improve the search direction under the guidance of swarm intelligence rather randomly.
4. Simulation results showed that the VNE algorithm we propose is better than what was offered in prior research in terms of reducing energy consumption and improving the request acceptance ratio and revenue-cost rate.

The remainder of this paper is organized as follows: in Section 2, we revisit the related works; the network model and performance metrics are presented in Section 3; in Section 4, we provide details of our proposed VNE algorithm; in Section 5, we compare our EBFO algorithm to other state-of-the-art algorithms; and the conclusions of the study are presented in the last section.

II. RELATED WORK
Given that VNE is an NP-hard problem, many scholars have performed meaningful research while focusing on finding an approximate optimal solution combined with nature-inspired
computations. For example, our previous work [18] proposed an improved particle swarm optimization algorithm, which addressed random factors, operated along the original direction, and changed the introduction of history factors to the search process. Furthermore, Zhang et al. [20], who improved three aspects of the classic genetic algorithm – population initialization strategy, mutation operation, and improvement operation – proposed a VNE algorithm based on a modified genetic algorithm. Liu et al. [21] leveraged the iterations and intelligence of artificial bee colonies to achieve an optimal solution for VNE. Zhang et al. [22] leveraged the niche particle swarm optimization technique to design a meta-heuristic algorithm to solve this problem. Cao et al. [23] developed an ant colony optimization algorithm of virtual network embedding. However, in terms of the algorithm structure, the swarm intelligence algorithms do not fully take into account the communication between the organisms in the process of foraging resulting from the limited food supply. In terms of algorithm realization, the strategies proposed by the algorithm to improve convergence speed or jump out of the local optimization promote the algorithm itself, rather than focus on VNE. These strategies do not consider the necessary energy-saving factors, such as the connections between networks and the underlying resource utilization. This drawback can be harmful in managing energy consumption and cannot guarantee the InPs’ long-term revenue.

Therefore, to reduce energy consumption and improve the long-term benefits of InPs, an increasing number of researchers have focused on the network topology because the management of network topology can make the utilization of underlying resources more effective. Bianchi and Presti [25] presented a two-stage VNE algorithm that focuses on the availability of physical resources and proposes a node-ranking algorithm based on a Markov chain. First, this algorithm maps virtual nodes to substrate nodes based on the node-ranking algorithm and then maps links along the shortest paths among the nodes. Cheng et al. [26] devised a topology-aware measure to rank nodes based on random walks and then devised a greedy algorithm that matches nodes in the VN to nodes in the substrate network according to node ranks. In addition, to approximate this problem, it devised a fast-convergent discrete particle swarm optimization. Gong et al. [27] devised a metric called global resource capacity to quantify the embedding potential of each substrate node and proposed an efficient heuristic VNE algorithm. However, these algorithms focused on substrate resource availability (i.e., using nodes with high available capacity first), not on their connection or the network topology structure, which would make the position of the mapped nodes scattered, and in this situation lead to the use of unnecessary substrate resources. Furthermore, at the beginning, these resources might obtain a higher request acceptance rate. However, as the substrate resources become occupied and links become segmented, when a VNR that requests a large resource arrives, they may not find an appropriate physical network to fit it, leading to a lower request acceptance rate and decreased resource utilization.

Many researchers set the corresponding energy-saving manager based on controlling the network topology to make more reasonable use of the underlying resources. For example, our previous work [19] considered the problem of scattered node mapping and hops of link mapping in traditional VNE embedding and proposed an energy-saving VNE algorithm based on a sliding region particle swarm. Reference [28] proposed a VNE algorithm employing Q-learning and curiosity-driven mechanism in considering other nondeterministic factors of avoiding falling into a local optimum. Zhu and Wang [29] modeled the VNE problem based on VN topology invariance and designed a modified ant colony optimization algorithm based on network topology decomposition. Jahani and Mohammad [30] modeled VNE as a multiobjective optimization problem and solved it by a non-dominated sorting genetic algorithm. Konglin and Zhihan [31] proposed a restricted differential evolution topology virtualization algorithm to achieve efficient mapping of a virtual network to a physical network in full virtualization. However, all this research sets the energy manager only in the mapping phase, leaving the optimization process to be executed without being energy-driven. As they work through the iterations, the topology of the VN’s physical hosted network, originally proposed under its energy manager, constantly changes along with the position of the nodes, which may cause the final optimal solution not to guarantee both energy efficiency and mapping success rate.

Therefore, keeping the VNE solution’s validity and energy savings in mind, this paper proposes an energy-driven enhanced BFO algorithm to solve the VNE problem.

III. PRELIMINARIES

A. THE INTRODUCTION OF NETWORK MODEL

1) SUBSTRATE NETWORK

We model the substrate network as an undirected graph $G_s = (N_s, E_s, S_s, G_{sm}, E_{se})$, where $N_s$ is the set of nodes, every node $n_s(n_{sc}, s_{so})$ with two attributes: $n_{sc}$ (capacity of CPU) and $s_{so}$ (turn on or off); $E_s$ is the set of links, where the link $e_s(e_{sl}, e_{so})$ also has two attributes: $e_{sl}$ (capacity of bandwidth) and $e_{so}$ (turn on or off). $S_s$ is the socialite of the SN, representing the most interactive node in the physical network (the details are described in section IV-B). $G_{sm}$ is the backbone network, and $E_{sm}$ is the remaining links of the network.

2) VIRTUAL NETWORK

Each virtual network is designated by a quintuplet $G_v = (N_v, E_v, S_v, G_{vm}, E_{ve})$, where $N_v$ and $E_v$ are the set of nodes and set of links, respectively. Each virtual node or link is associated with the constraint of the requested capacity: CPU and bandwidth. $S_v$ is the socialite of the VN; $G_{vm}$ is the backbone network; and $E_{vm}$ is the remaining links of the network. Furthermore, VNR can be described as VNR($ta, td$),...
where \( t_a \) stands for the request arrival time and \( t_d \) stands for the duration of the VN request hosting in the SN.

### B. THE PERFORMANCE METRICS

The goal of this paper was to reduce energy consumption while ensuring the request acceptance rate and the revenue-cost ratio, i.e., to propose an energy-saving VNE strategy while ensuring the validity of the proposed solution.

**Energy**

The energy spent by allocating VNR to physical resources can be defined as follows:

\[
E(\text{VNR}) = E(N_v) + E(e_v) \\
= \sum_{n_v \in N_v} E(n_v) + \sum_{e_v \in E_v} E(e_v)
\]

where \( E(N_v) \) and \( E(e_v) \) represent the energy consumption of the physical nodes and links, as defined in Eqs. (2) and (3), respectively.

\[
E(n_v) = \begin{cases} 
    n_v + (n_v \text{, max} - n_v - n_v, \text{base}) \times \tau, & \text{if the selected physical node } n_v \text{ is open} \\
    0, & \text{otherwise}
\end{cases}
\]

\[
E(e_v) = \begin{cases} 
    e_v, \text{base}, & \text{if the selected physical link } n_v \text{ is open} \\
    0, & \text{otherwise}
\end{cases}
\]

where \( n_v, \text{base} \) is the basic power consumption of physical node, \( n_v, \text{max} \) is the full-load power consumption of physical node, \( \tau \) represents the CPU load rate of substrate node, and \( e_v, \text{base} \) indicates the link energy consumption, which is generally constant.

**Acceptance**

The success mapping rate, i.e., the ratio between the number of virtual networks successfully mapped and the total number of virtual network requests within a certain period of time, can be defined as follows:

\[
R_{\text{accept}} = \frac{\sum_{i=1}^{n_\theta(T)} \sigma(\text{VNR}_i)}{n_\theta(T)}
\]

where \( n_\theta(T) \) indicates the number of requests reached VNRs during a \( T \) period, \( \sigma(\text{VNR}_i) \) indicates whether the \( i \)-th VNR is mapping successfully. If yes, the value is 1; for others it is 0.

**Revenue-cost**

The revenue-cost ratio represents the revenue generated by consuming unit resources, and it can be defined as follows:

\[
\frac{R}{C} = \frac{\sum \sigma(G_v)}{\sum C(G_v)}
\]

where \( R \) represents revenue (similar to most prior work, we take the CPU capacity revenue and bandwidth capacity of VNR as its revenue): 

\[
R(G_v) = \bar{\beta} \sum_{n_v \in N_v} \text{cpu}(n_v) + \sum_{e_v \in E_v} \text{bw}(e_v)
\]

\( C \) represents energy cost for mapping the \( i \)-th VNR, and it can be defined as follows:

\[
C(G_v) = \beta \sum_{n_v \in N_v} \text{cpu}(n_v) + \sum_{e_v \in E_v} N \cdot \text{bw}(e_v)
\]

where \( \beta \) and \( \bar{\beta} \) represent the linear parameters, \( n_v \) and \( e_v \) represent the virtual node and link, respectively, and \( N \) stands for how many substrate nodes have been powered up.

### IV. AN ENERGY-DRIVEN VNE ALGORITHM BASED ON ENHANCED BFO

BFO is a novel bionic optimization algorithm inspired by the social foraging behavior of \( E. \coli \). This algorithm was first proposed by Passion in 2002 [33]. BFO is mainly governed by four processes: swarming, chemotaxis, reproduction, and elimination/dispersal. The originality of our work is in the leveraging of the enhanced BFO to develop an energy-driven VNE solution. However, first, we will provide a brief introduction to the BFO basics.

#### A. BFO BASICS

In the BFO algorithm, the objective is to find the optimum value of the nutrient gradient \((J(\theta))\), where \( \theta \) stands for the position. The virtual bacteria in the group are actually one trial solution (which may be called a search agent) that moves on the functional surface to locate the global optimum, and the bacteria are made up of individual bacteria (cells). Here, we identify the cell as the bacterial position vector. The details are as follows:

**Swarming**

A group of \( E. \coli \) cells arrange themselves in a traveling ring to move up the nutrient gradient by sending signals (including cell-to-cell and bacteria-to-bacteria) to form several intricate and stable spatiotemporal patterns (swarms). Fig. 2- (c) depicts this behavior in the white circle.

**Chemotaxis**

is a foraging behavior that implements a type of optimization where bacteria try to climb up the nutrient concentration. Biologically, a cell can move in two different ways: swimming (Fig. 2- (a)) and tumbling (Fig. 2- (b)) via flagella.

**Reproduction**

The least healthy bacteria eventually die while the healthier bacteria asexually split into two bacteria,
A mapping strategy was designed mimicking the interaction between bacteria, identified as SBR_E, which was then leveraged to design an energy manager for the optimization process. This approach enhanced the BFO, allowing the development of an energy-driven VNE algorithm identified as EBFO.

**TABLE 1. Physical resource priority level.**

| Node/Link state | Whether Meet the resource requirements | On or off | Priority |
|-----------------|--------------------------------------|-----------|----------|
| State 1         | F                                     | \         | 0        |
| State 2         | T                                     | F         | 1        |
| State 3         | T                                     | T         | 2        |

which are then placed in the same location. This approach keeps the swarm size constant.

**Elimination/dispersal** events can take place in such a fashion that all the bacteria in a region are killed or a group is dispersed into a new location. To simulate this phenomenon in EBFO, some bacteria are liquidated at random with a very small probability, while the new replacements are randomly initialized within the search space.

Based on the above information, it is known that BFO has a complete structure and low coupling between modules. However, its initial position and direction of chemotaxis are chosen randomly, which can slow the algorithm’s convergence speed. In addition, if it is employed to approximate the optimal solution of the VNE problem, an energy manager is needed to monitor the energy consumption throughout the whole optimization process to use the underlying resources rationally. With the goal of addressing these challenges, a mapping strategy was designed mimicking the interaction between bacteria, identified as SBR_E, which was then leveraged to design an energy manager for the optimization process. This approach enhanced the BFO, allowing the development of an energy-driven VNE algorithm identified as EBFO.

**B. A MAPPING STRATEGY BY MIMICKING THE INTERACTION BETWEEN BACTERIA: SBR_E**

This section focuses on the connection between networks and applies the cell-to-cell signaling behavior to the process of finding the SN to host the VN, leading toward the development of a mapping strategy SBR_E, which is used to build the candidate set of the VNE solutions. First, the topology information of VN is extracted and then the VN is mapped to the SN following the extracted topology information. To facilitate the description, we prioritize the physical nodes and links as follows:

1) **EXTRACT THE SBR**
   First, the main information of the virtual network is extracted. The value of the link’s bandwidth is taken as the weight (in this instance, we set the weight as the inverse rate with the value), and the Prim algorithm is employed to construct a minimum spanning tree aiming to extract the VN’s backbone network. In Fig. 3, the light blue square represents the virtual node, the line between them represents the link, and the value represents its bandwidth capacity. The extracted backbone network is presented on the right of the diagram marked by (b), termed $G_{vem}$, which consists of the blue solid lines and the nodes. The red dotted lines represent the remaining links, termed $E_{vem}$. In this way, we separate the main topology information (backbone network) from the secondary information (remaining links) of the virtual network and use different mapping methods to map them.

   Then, by mimicking the cell-to-cell interaction, a socialite is selected from the bacteria, from which we can obtain the fastest convergence speed to spread the information over the bacteria. In the virtualization environment, a node should be selected from the backbone network to be the socialite that has the shortest distance from other nodes. As shown in Fig. 4-(a), the yellow node in the center is the socialite of this VN, termed $S_v$. Therefore, socialite is always the node with the greatest number of degrees.

   After extracting the main topology information, and selecting the socialite. We map the $G_v$ onto the $G_s$ in the order of socialite $\rightarrow$ backbone network $\rightarrow$ remaining links. In this topology-aware mapping method, the virtual nodes and links of the backbone network are mapped in the same period and considered as a whole structure, which will reduce the hops of link mapping, further saving the energy and using different mapping methods for the main and secondary information of the network can effectively improve link utilization and mapping success rate. The mapping process is described below. For further details, see algorithm 1.

   2) **MAP SBR** ($G_v$) **ONTO THE** $G_s$

   a: **MAP** $S_v$ **ONTO** $S_s$

   The node with the strongest communication ability is calculated as the socialite. Then, the physical nodes meeting
Algorithm 1 SBR_E Mapping Algorithm

input: the socialite of the bacteria—\(S_v, G_v, G_s\)
output: the position of bacteria (mapping solution)

1. while the quantity of unmapped nodes in \(G_{vm}\) ! = 0
2. do:// mapping \(G_{vm}\) onto \(G_{sm}\)
3. take the \(S_v\) as root node:
4. reorder the \(S_v\) and \(S_v\)’s adjacent links by their bandwidth in a descending order,
5. if \(S_v\)’s degree > \(S_v\)’s
6. Employ L2S2 principle to map these links,
7. if the available resources of the nodes on the link’s other side may satisfy the resource constraint,
8. map these nodes on them;
9. else keep extending outward to the next hop,
10. pick the node with the largest weight to allocate this nodes
11. end if
12. else
13. merge these links and map them on the \(S_v\)’s adjacent links
14. extend to the node on the other end of the merged link, and from its neighbors, pick the node with the largest weight to allocate this node
15. if the out-degree of \(S_v\)’s neighbor node \(N_v^2\) ! = 0:
16. for \(N_v^2\) do:
17. take the \(N_v^2\) as root node:
18. return (4)
19. end for
20. end do
21. employ the Dijkstra’s Shortest Path Algorithm to map the remaining links //mapping \(E\) onto \(E\)

the CPU resource constraints with the highest priority were selected as the candidate set.

b: MAP \(G_{vm}\) ONTO \(G_{sm}\)
After selecting \(S_v\), we set \(S_v\) as the root, and the link’s bandwidth is taken as the weight (here, we set the weight in direct ratio with the bandwidth) and borrow the idea of the breadth-first search algorithm to reorder the backbone network tree (marked as \(G_{vm}\)) by hierarchy. That’s to say, the reordering process must be the first to select the node closer (the one with fewer hops) to the socialite. In addition, As shown in Fig. 4- (a), the number in the square (node) is the order in which the node is reordered, i.e., socialite is the first selected node, the neighbor node with the largest bandwidth of \(S_v\) is the second and so on. To make the representation easier to understand, we redraw Fig. 4- (b) to show the reordered \(G_{vm}\).

Then, according to \(G_{vm}\)’s new order, we find the physical network \(G_{sm}\) to allocate it. In this phase, we expand the mapping around the selected physical node \(S_v\), which hosts \(S_v\), and consider the nodes and links mapping at the same time by hierarchy. As shown in Fig. 4-(b), the reordered \(G_{vm}\) is divided into three parts based on the number of hops from socialite. We place these nodes and links around \(S_v\) from near to far. For mapping \(S_v\)’s neighbor nodes and links between them (in 3), we allocate these adjacent links first: if \(S_v\)’s degree > \(S_v\)’s degree, the available bandwidth of \(S_v\)’s adjacent links can satisfy the resource constraint of \(S_v\)’s adjacent links. We follow the large-to-large and small-to-small (L2S2) principle [34] to map \(S_v\)’s adjacent links. Then, we further determine whether the selected node at the other end of the link satisfies the CPU resource constraint of the virtual node; if yes, map them; otherwise, extends outward to the next hop and picks the resources with the largest weight to allocate the node that failed to match. Then, the next hop’s nodes and links (in 3) should be mapped by setting the \(S_v\)’s adjacent node as the new root and repeating the above operation until all the nodes and links are mapped. In this phase, the links and the nodes with high priority (in the powered state and satisfy the constraint) will be selected first; moreover, after such processing, the selected physical hosted network will be as compact as possible, which will greatly reduce unnecessary energy consumption.

c: MAP \(E\) ONTO \(E\)
The remaining links seem to be secondary information of the network. The number of links is relatively small, and the information interaction between a few nodes at the edge is carried. Therefore, we simply employ Dijkstra’s Shortest Path Algorithm to find the physical links (\(E\)) and place the remaining links \(E\).

C. ENERGY-DRIVEN VNE ALGORITHM BASED ON BFO: EBFO
Based on the above, we know that to reduce the energy consumption of VNE, prior research has put forward many strategies, but most of them focused on physical resource availability or just set the energy-saving manager on the mapping phase. In this section, to approximate the optimal VNE solution, we synthetically consider the information interaction among the bacterial flora to design the optimization process, leverage the SBR_E algorithm to monitor this process’s energy consumption, and present an optimal solution for the VNE problem.

When seeking a higher concentration of nutrients, the bacterial flora has two kinds of information interactions: ① cell-to-cell signaling and ② bacteria-to-bacteria signaling. Considering the latter, EBFO added a corrected value for the objective function according to the cell aggregation. In this section, we provide a comprehensive review of these two signaling processes and design an energy manager to control energy consumption globally. In addition, to perfect the algorithm’s execution process, we enhance the BFO algorithm using the characteristics of the network, such as chemotaxis, reproduction, and elimination/dispersal, so the optimal embedding solution can be found. In the following subsections, we will describe the details of EBFO.
1) SWARMING AND NUTRIENT GRADIENTS

The goal of this paper was to find an effective and energy-saving embedding scheme for the VNE problem. Therefore, we first define the objective function (to be minimized) as energy saving:

\[ J(i, j, k, l) = f(E) \] (8)

However, in the natural world, the nutrient concentration is not static. It is influenced by bacteria-to-bacteria signaling, including attractant and repellant signaling here, we identify it as the corrected value \( J_{\theta} \) to dynamically control the aggregation of the bacterial flora. Similarly, in the solution space of VNE, if the proposed solutions are too aggregate (similar), the VNE solution will be vulnerable to premature convergence. Therefore, we add on the cell-to-cell attractant–repellant profile \( J_{\theta} \) to the objective function and renew it with the help of the following equation:

\[
J(i, j, k, l) = J_E + J_{\theta}(\theta, P(j, k, l))
\] (9)

\[
J_{\theta}(\theta, P(j, k, l)) = \sum_{i=1}^{s} J\theta(\theta, P(j, k, l))
\]

\[
= \sum_{i=1}^{s} -\rho \exp(-\omega) \sum_{m=1}^{d} (\theta_m - \theta^i_m)^2 \] (10)

\( J_{\theta} \) represents the effect of bacteria-to-bacteria signaling on the nutrient gradient in the current position, which is added to the actual objective function to present a time varying, objective function. \( S \) is the total number of bacteria in the flora, \( d \) is the number of variables to be optimized, which are present in each bacterium, and \( \theta^i = (\theta^i_1, \theta^i_2, \ldots, \theta^i_n)^T \) is a point in the \( d \)-dimensional search domain. \( \Theta \) represents a command instructing to return a value of 0 if both sides are equal, or return 1 if they are not. \( \rho \) represents the height of the repellent effect (magnitude of its effect) and \( \omega \) represents the width of the repellent effect. In addition, we set \( J_{\theta} \) as a negative number because in chemotaxis, the iterative update process will reduce the width of the repellent effect. In addition, we set \( \omega \) equal, or return 1 if they are not.

2) INITIALIZATION

In this paper, we use an enhanced BFO algorithm to solve the optimization problem. Here, we take the position of a particle denoted by \( \theta^i = (\theta^i_1, \theta^i_2, \ldots, \theta^i_n)^T \) as the solution of node mapping. \( \theta^i_1 \) can be assumed as the solution of \( S_i \) mapping, and the \( i^\text{th} \) position of \( \theta^i_n \) can be assumed as the solution of the virtual node \( vi \) mapping.

The initial position of the bacteria in the BFO algorithm is randomly selected. However, in VNE, if we overuse the bottleneck resources of the substrate network, it may fragment the network resources and thus prevent the substrate network from accepting a larger VN network request. If the position is very disposal, the distance between nodes would be long and would require many bandwidth resources. Moreover, since the position vector of bacteria corresponds to a node mapping solution, if we randomly generate one or more dimensions of this vector, it may violate the capacity constraint or the connectivity constraint of virtual links in the linking mapping stage, which may slow down the convergence speed of our algorithm. Therefore, we mimicked the information interaction between the bacterial group and leveraged SBR_E to manage this process’s energy. Based on this, we present an SBR preferred selection strategy for position initialization. This strategy employs network topology information (SBR) to guide the process of selecting the bacterial flora’s initial position and prioritizes the substrate resources in the on state that meet the constraints. Such a strategy can prevent the final solution from having low validity or high energy consumption caused by the nodes’ overuse or overdispersion. A detailed description is provided in algorithm 2.

**Algorithm 2 SBR Preferred Selection Strategy for Position Initialization**

**Input:** \( G_V, G_S \)

**Output:** the initialization of bacteria

1. Calculate the \( G_V \)’s socialite: \( S_i \); 
2. Randomly select \( N \) physical nodes, which make up the socialite set of bacteria in the flora, termed as \( S = \{s_1, s_2, \ldots, s_N\} \); 
3. For \( S_i \) in \( S \): //initialize the position of bacteria in the flora
4. Call the SBR_E, identify the result as the initial position of this bacteria;
5. End for;
6. Employ Formula (9) to compare the position’s nutrient gradient of bacteria in the flora and record the position with the highest gradient;

The SBR preferred selection strategy for initialization shares the same idea with SBR_E mapping presented in section IV-(B). First, \( N \) physical nodes are randomly selected from the substrate network as the socialite of different bacteria in the flora, which is the \( G_S \)’s \( S_i \) to allocate \( S_i \). Then, SBR_E is employed to arrange the other nodes and links.

\( |N_i| \) represents the number of physical nodes, \( |E_i| \) represents the number of links in the backbone network, and \( |E_{se} \) represents the number of remaining links. Algorithm 2’s time complexity can be calculated as the time complexity of mapping the backbone network plus the remaining links, i.e.,

\[ \alpha(|N_i| + |E_i| + |E_{se}| \cdot |N_i|^2) \]

3) CHEMOTAXIS

This section is similar to the initialization, where we leverage SBR_E to ensure that this iterative update process is energy driven and design an SBR preferred selection strategy for position update in algorithm 3. When the nutrient concentration of the living environment is changed, we set the socialite first aware of this change and update its position. Then, the message is transmitted to all bacteria according to the following order: socialite → main network → residual link.
updating them in the process. This pushes all bacteria to move to the position with higher nutrient concentration. Such a strategy could control the energy consumption of the whole process, help simplify the optimization process of swarm intelligence, and reduce the complexity of the problem.

In computational chemotaxis, the movement of the bacteria may be represented by the following:

$$\theta^i(j + 1, k, l) = \theta^i(j, k, l) + C(i) \varphi(j)$$

where $\theta^i(j, k, l)$ represents the position of each member in the population of the $S$ bacteria at the $j$th chemotactic step, $k$th reproduction step, and $l$th elimination-dispersal event. $C(i)$ is the size of the step taken in the random direction ($\varphi(j)$) specified by the tumble (run length unit).

Such a strategy may clearly slow down the convergence speed. To address this challenge, we employ the swarm intelligence to guide the socialite’s updating and change $\varphi(j)$ under the guidance of $\theta^i_{ps}$ and $\theta^i_{gs}$. Here, $\theta^i_{ps}$ is the position with the highest nutrient gradient found so far for the $i$th bacteria, and $\theta^i_{gs}$ is the best position in the swarm. Therefore, we renew the socialite’s velocity ($V^i_{is}^{\text{new}}$) and its position ($\theta^{i\text{new}}_{is}$) with the help of the following equations:

$$V^i_{is}^{\text{new}} = \omega d \cdot V^i_{is}^{\text{old}} + [c_1 \cdot r_1 \cdot (\theta^i_{ps} \ominus \theta^{i\text{old}}_{is}(j + 1, k, l))] + [c_2 \cdot r_2 \cdot (\theta^i_{gs} \ominus \theta^{i\text{old}}_{is}(j + 1, k, l))]$$

$$\theta^{i\text{new}}_{is}(j + 1, k, l) = \theta^{i\text{old}}_{is}(j, k, l) \otimes V^i_{is}^{\text{new}}$$

where $r_1$ and $r_2$ stand for the two random variables that are uniformly distributed in $[0, 1]$. $\omega_d$ stands for the inertia weight, $c_1$ stands for the cognition weight, and $c_2$ stands for the social weight. Generally, $\omega_d$, $c_1$, and $c_2$ meet the following constraints: $0 < \omega_d \leq P_c < P_s < 1$ and $P_l + P_c + P_s = 1$.

$\ominus : \theta_1 \ominus \theta_2$ return the difference of the two socialite mapping solutions, the resulting value is 1 if $\theta_1$ and $\theta_2$ have the same values, and 0 otherwise.

$\oplus : \theta_1 \oplus \theta_2$ return the numerically superimposed direction of bacteria’s socialite. If the result value is 1, tumble toward the direction of $\theta^i_{ps}$ or $\theta^i_{gs}$ in the probability; otherwise, they keep running without changing the direction.

$\otimes : \theta^{i\text{old}}_{is} \otimes V^i_{is}^{\text{new}}$ indicate updating the socialite’s position. Its operation rule is this – if the value of $V$ equals 1, the socialite guided by the swarm intelligence chooses the best direction, singles out the node closest to socialite, and meets the resource constraints with the highest priority; otherwise, it keeps running and selects a neighbor node of the socialite with the highest priority. As shown in the Fig. 5-(a), the pure yellow node represents the initial position of the socialite, and the yellow node with the aperture represents the updated position.

After the socialite’s update, we continue to update the backbone network and the remaining links, as shown in the bottle of Fig. 5-(b). The bacteria’s initial position is within the broken black circle. Substrate nodes in the off state are shown as transparent, and nodes in the on state are light blue. Dark blue represents the mapped nodes, the bold black line represents the backbone network’s links, and the red line represents the remaining links.

If the number of bacteria in the flora is $n$, then algorithm 3’s time complexity can be calculated as $o(|N_i| + |E_i| + |E_{se}| \cdot |N_i|^2)$.

4) REPRODUCTION AND ELIMINATION/DISPERSAL
To improve the convergence speed of the algorithm, the algorithm enters the reproduction stage after the iteration and updates until it reaches the predetermined number of times. The corresponding VNE solution whose nutrient gradient is in the top 50% will be copied into two and continue iterative update driven by energy awareness. The last 50% will be eliminated.

To avoid the algorithm falling into local optimization and premature convergence, the algorithm enters the elimination/dispersal after jumping out of the cycle of reproduction. Some bacteria are liquidated at random, while the new VNE solutions are randomly initialized leveraging the SBR initialization method within the search space. Then the new floral bacteria return to the chemotaxis, continue iteration, and update constantly until the given number of times is reached, recording the current optimal solution, which is the VNE solution proposed by the EBFO for the VNR request.

D. ALGORITHM DESCRIPTION
The realization process of EBFO was introduced in the previous three sections. This section gives an overall description of it. The pseudocode as well as the flow-chart (Fig. 6) of the complete algorithm are presented below.
Algorithm 3 SBR Preferred Selection Strategy for Position Update

> input: $G_V, G_S$
> output: the $\theta_{es}$ of flora

(1) **For** $S_i$ in flora:
(2) employ Formula (6) to calculate the updated direction $V$;
(3) if $v = 1$
(4) updating toward $\theta_{ps}^i$ or $\theta_{gs}$ in the probability
(5) take $S_i$ as root node
(6) if the root has the neighbor nodes that satisfy its constraint;
(7) calculate the sum of the node’s available resource of CPU and the link’s bandwidth between the node and the root as the weight.
(8) select the node $(S_i)$ with the largest weight to host $S_i$
(9) return(6)
(10) end if
(11) else
(12) $S_i$ keep running and select a neighbor node of the socialite with most available CPU resources;
(13) end if
(14) end for
(15) employ Formula (9) to compare the position’s nutrient gradient of bacteria in the flora
(16) update $\theta_{ps}^i$
(17) update $\theta_{gs}$

**Parameters:**

**[Step 1]** Initialize parameters $p, S, N_c, N_s, N_{re}, N_{ed}, P_{ed}, i, \theta, S$

**Algorithm:**

**[Step 2]** SBR_E preferred selection strategy for position initialization

**[Step 3]** Elimination-dispersal loop: $l = l + 1$

**[Step 4]** Reproduction loop: $k = k + 1$

**[Step 5]** Chemotaxis loop: $j = j + 1$

**[Step 6]** SBR_E preferred selection strategy for position update

**[Step 8]** If $N_c < j_g$, go to step 5. In this case continue chemotaxis since the life of the bacteria is not over.

**[Step 7]** Reproduction:

[a] For the given $k$ and $l$, and for each $i = 1, 2, \ldots, S$, let, $J(i, j, k, l) = J_E + J_{\theta}$

[b] The $S_i$ bacteria with the highest $J$ values die, and the remaining $S_r$ bacteria with the best values split.

**[Step 8]** If $N_{re} < k$, go to step 4.

**[Step 9]** Elimination-dispersal: For $i = 1, 2, \ldots, S$ with probability $P_{ed}$, employ SBR initialization to make up the bacteria which will be eliminated. If $1 < N_{ed}$, then go to step 3; otherwise end.

**[Step 10]** Record the current global optimal solution as the best mapping scheme of this VNR.

V. PERFORMANCE EVALUATION

In this section, we compare our proposed algorithm known as EBFO to the prior algorithms. In subsection A, we describe our simulation settings, and in subsection B we present our evaluation results.

A. A EVALUATION SETTINGS

1) NETWORK TOPOLOGY

Similar to our previous studies [28], we use the GT-ITM tool to generate the bacterial search topologies of the substrate network and virtual network. The number of substrate network nodes predesigned in this experiment is 50, and the probability of the nodes connecting to each other is assumed to be 0.5.

For the capacity settings, the CPU capacity is uniformly distributed between 50 and 100, and the bandwidth is also uniformly distributed between 50 and 100. Between each pair of substrate nodes, it has a probability of 0.5 to connect each other. The number of nodes in a VNR has a uniform distribution of $U[2, 10]$. We follow the literature [28], and each virtual node’s CPU and bandwidth capacity and requirement obeys the uniform distribution of $U[0, 25]$. The process of VNR arrival can be simulated as a Poisson process, and its average arrival rate is set to 5 VNRs per 100 time units. Its residence time obeys an exponential distribution with an average value of 500 time units. To map a sequence of 2000 VN requests, our simulation may last for approximately 40,000 time units. A time window is equal to 2000 time units. We run all of our simulations in 20 time windows and collect the experimental data after 5 time windows.

2) PARAMETER SETTINGS

The values of $p_{max}, P_b,$ and $P_n$ are assigned 300, 150, and 1.5 $W$ per CPU unit, respectively, following previous work [28] The initial number of bacteria in flora $N$ is set to 12, and the threshold of maximum count of iteration $N_c, N_s, N_r$ are set to 5, 3, and 2. According to our experience, the initial values of $\omega_1, \omega_2$ and $c_2$ are set to 0.1, 0.2, and 0.7, respectively. $d$ and $\omega$ are set to 0.05.

3) COMPARISON

Table 2 presents an evaluation of the three algorithms. We compare our algorithm (EBFO) to three prior algorithms – MO_NPSO [22], MCRR [25], EE_CTA [30], and the discrete BFO. Note that we assume that our proposed algorithm does not support path splitting, and the virtual nodes of different VN requests can be mapped onto the same substrate node, but the virtual nodes in the same VN request cannot share the same substrate node.
TABLE 2. Compared algorithms.

| Notation | Algorithm |
|----------|-----------|
| EBFO     | An energy-driven VNE algorithm based on enhanced BFO, aiming at saving energy and effectively improving VNRs acceptance rate and the revenue-cost ratio. |
| MO_NPSO  | An VNE algorithm based on enhanced PSO algorithm. It leverages niche particle swarm optimization technique to design a meta-heuristic algorithm to find a VNE solution with low cost and energy. |
| MCRR     | A two-stage VNE algorithm that focuses on the availability of physical resources, proposing a node ranking algorithm MCRR based on Markov chain. |
| EE_CTA   | A meta-heuristic VNE algorithm sets the energy manager only in the mapping phase. It models VNE as a multi-objective optimization problem and solving it by Non-dominated Sorting Genetic Algorithm. |
| BFO      | A VNE algorithm based on discrete BFO, with other parameters set the same as EBFO, it was designed to prove the effectiveness of EBFO over BFO. |

FIGURE 6. The flow-chart of EBFO.

B. EXPERIMENTAL RESULTS

We summarize our experimental results in terms of energy saving, request acceptance rate, revenue-cost ratio, and running time.

1) ENERGY

The energy comparison results are presented in Fig. 7. We can observe that while ensuring the effectiveness of the solution, EBFO still has more energy than MO_NPSO, MCRR, and EE_CTA. That difference is observed because although MO_NPSO sets the energy saving as the goal, it does not fully take into account the communication between the particles, i.e., it ignores the necessary energy-saving factors – the connections between networks and the underlying resource utilization. Therefore, its energy saving effect is very limited. MCRR pays attention only to finding nodes that have more available resources. As the underlying resources are opened during the embedding process, this greedy algorithm will open increasingly more resources, resulting in the waste of resources. As a meta-heuristic algorithm, EE_CTA iteratively finds the optimal VNE solution with the goal of saving energy. However, as the situation of the substrate network changes over time, the node position will be increasingly scattered, and the number of link map hops will increase, which leads to a continuous increase in energy consumption. In contrast, EBFO sets energy savings as a global goal. It synthetically considers the information interaction among the bacterial flora to devise a topology-aware mapping strategy marked as SBR_E to build the candidate set of the VNE solutions. Then, it leverages SBR_E to manage the optimization process and approximates the optimal VNE solution. In this way, after the socialite finds a better location, we reanalyze the current underlying network environment and map VNR in the order of socialite → backbone network → remaining links. In this process, we preferentially select the underlying resources that have been opened for mapping, and the nodes and links of the backbone network are mapped as a whole structure tightly around the socialite. In this way, the distance between nodes can be controlled more strictly, and the number of link hops will be greatly reduced. From Fig. 8-11, we can note that EBFO uses fewer substrate nodes and links than other algorithms and has a higher utilization rate of substrate nodes and links. This helps demonstrate why EBFO achieves more energy-efficient VNE solutions. In addition, although the BFO algorithm has a lower energy consumption, it has a very low quality VNE solution and thus cannot be a stable VNE problem-solving strategy.

2) REQUEST ACCEPTANCE RATE COMPARISON

The results of the average revenue comparison are shown in Fig. 7. It is evident that from the point of view of the request acceptance rate, EBFO is much better than other algorithms.
From the results, we can see that the request acceptance rate of BFO is less than 65%; for MO_NPSO and MCRR, it is approximately 80%; for EE_CTA, it is approximately 85%; and for EBFO, it is 95%. That means that our algorithm can at least obtain up to a 10% higher request acceptance rate than other prior algorithms and can obtain up to a 35% higher request acceptance rate than the original BFO. This is because in the mapping process, EBFO insists on prejudging the selected physical network, guaranteeing that the capacity constraints of VNRs are satisfied. Although its solution continues to change with iteration, it leverages SBR_E to control the topology of the solution to ensure that nodes and links are available, further ensuring the validity of its solution. However, MCRR that maps the VNRs onto the substrate network focuses only on the underlying resources, which have more available resources, and the prophase undergoes a high request rate. As an increasing number of physical nodes are powered up, the corresponding physical links are divided, and when a VNR, which requires plenty of resources, is dynamically reached, it may not find the substrate network to host it, causing the request acceptance rate to decrease. MO_NPSO and EE_CTA, these meta-heuristic algorithms, have an independent optimization process without a resource manager. Therefore, guaranteeing the VNE solution that the algorithms calculated is the most energy-efficient. These algorithms have to sacrifice a certain solution’s quality. Because the node position and the situation of the substrate network change all the time, in the pursuit of an energy-saving solution, the connectivity between nodes will be difficult to guarantee at the same time. This difficulty means that the most energy-efficient solution may be proposed at the expense of the solution’s quality. Therefore, these solutions cannot be as good as EBFO in obtaining high request acceptance rate.

3) REVENUE-COST RATIO COMPARISON

The results of the average revenue comparison are shown in Fig. 8. We can observe that from the point view of revenue-cost ratio, EBFO is much better than the other algorithms. From the results, we can see that the request acceptance rates of MO_NPSO, MCRR, EE_CTA, and BFO are 0.5, 0.6, 0.55, and 0.68, respectively. This means that EBFO can obtain between 10% and 30% higher revenue-cost ratios than the other algorithms. This is because the excitation process of EBFO is energy driven and has achieved an excellent energy-saving effect, after chemotaxis, reproduction, and elimination dispersal find the optimal solution for the VNE problem. In the whole process, the distance between nodes is strictly controlled by the energy manager - SBR_E, which ensures the least number of hops between nodes when the virtual network is being mapped to the physical network, so the improvement effect of the revenue-cost ratio is remarkable.
4) RUNNING TIME COMPARISON

The average running times of the algorithms for mapping a VNR are shown in Fig. 12. From this figure, we see that BFO and EE_CTA (the two heuristic algorithms) consume more time than EBFO. The MCRR, which uses linear programming, has a running time that is less than EBFO. The running time of the EBFO algorithm is slightly more than that of the MO-NPSO algorithm, while it is worth achieving a higher request acceptance rate, revenue-cost ratio gain, and energy savings. However, it is still a large improvement over the original heuristic algorithms. The reason lies in the following: (1) EBFO has the ability of swarm intelligence’s parallel operation; (2) EBFO can significantly improve the convergence speed of the algorithm by adding swarm intelligence guidance to the chemotaxis; (3) EBFO leverages SBR_E to design SBR_E initialization and SBR_E chemotaxis methods, updating socialite first and then driving the whole bacteria to update, instead of updating all dimensions of the solution in the traditional group intelligence method, which reduces the complexity of the problem and significantly reduces running time.

VI. CONCLUSION

In this paper, we proposed an energy-driven algorithm based on BFO (EBFO) to solve the problem of VNE. Specifically, a mapping algorithm (SBR_E) based on an information interaction mechanism was designed to build the candidate set of the VNE solutions. Then, it leveraged it to manage the energy consumption of the optimization process. Finally, the optimal VNE solution is found. The comparative results prove that the proposed method can improve the performance of the system in terms of conflicting objectives by reducing energy consumption, improving the request acceptance rate, and improving the revenue-cost ratio. However, to ensure a high mapping success rate, the algorithm has to repeatedly examine the resource constraints, which affect the running time to some extent. This issue warrants further study.

REFERENCES

[1] B. A. M. Souza, G. R. Mateus, and F. S. H. de Souza, “Compact and extended formulations for the virtual network embedding problem,” Electron. Notes Discrete Math., vol. 64, pp. 205–214, Feb. 2018.

[2] C. K. Dehury and P. K. Sahoo, “DYVINE: Fitness-based dynamic virtual network embedding in cloud computing,” IEEE J. Sel. Areas Commun., vol. 37, no. 5, pp. 1029–1045, May 2019.

[3] S. Haeri and L. Trajkovic, “Virtual network embedding via Monte Carlo tree search,” IEEE Trans. Cybern., vol. 48, no. 2, pp. 510–521, Feb. 2018.

[4] B. Levie, F. Monti, X. Bresson, and M. M. Bronstein, “CayleyNets: Graph convolutional neural networks with complex rational spectral filters,” IEEE Trans. Signal Process., vol. 67, no. 1, pp. 97–109, Jan. 2019.

[5] K. Hejja and X. Hesselbach, “Online power aware coordinated virtual network embedding with 5G delay constraint,” J. Netw. Comput. Appl., vol. 124, pp. 121–136, Dec. 2018.

[6] F. Dong, X. Guo, P. Zhou, and D. Shen, “Task-aware flow scheduling with heterogeneous utility characteristics for data center networks,” Tsinghua Sci. Technol., vol. 24, no. 4, pp. 400–411, Aug. 2019.

[7] Y. Yuan, C. Wang, S. Peng, and K. Sood, “Topology-oriented virtual network embedding approach for data centers,” IEEE Access, vol. 7, pp. 2429–2438, 2019.

[8] M. K. Haghani, B. Bakhshi, and A. Capone, “Multi-objective embedding of software-defined virtual networks,” Comput. Commun., vol. 129, pp. 32–42, Sep. 2018.

[9] A. Jahanian and L. M. Khanli, “CaTa-VN: Coordinated and topology-aware virtual network service provisioning in data centers network,” in Proc. ICCKCE, Mashhad, Iran, 2017, pp. 353–358.

[10] G. Raidl and J. Puchinger, “Combining (integer) linear programming techniques and metaheuristics for combinatorial optimization,” Hybrid Metaheuristics, vol. 114, pp. 51–62, 2008.

[11] J. F. Botero, X. Hesselbach, and M. Duelli, “Energy efficient virtual network embedding,” IEEE Commun. Lett., vol. 16, no. 5, pp. 756–759, May 2012.

[12] I. Houidi, W. Louati, W. Ben Ammar, and D. Zeghlache, “Virtual network provisioning across multiple substrate networks,” Comput. Netw., vol. 55, no. 4, pp. 1011–1023, Mar. 2011.

[13] R. Mijumbi, J. Serrat, J.-L. Gorricho, and R. Boutaba, “A path generation approach to embedding of virtual networks,” IEEE Trans. Netw. Service Manage., vol. 12, no. 3, pp. 334–348, Sep. 2015.

[14] Y. Wang, Q. Hu, and X. Cao, “A branch-and-price framework for optimal virtual network embedding,” Comput. Netw., vol. 94, pp. 318–326, Jan. 2016.

[15] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in Proc. Neural Netw., Perth, WA, Australia, 1995, pp. 1942–1948.

[16] D. Karaboga, “Artificial bee colony algorithm,” Scholarpedia, vol. 5, no. 1, pp. 6915, 2010.

[17] T. Stutzle and M. Dorigo, Ant Colony Optimization. Cambridge, MA, USA: Bradford, 2004.

[18] Y. Hu and L. Zhuang, “Applying an improved particle swarm optimization algorithm in virtual network mapping,” Comput. Eng. Sci., vol. 11, no. 36, pp. 2169–2173, 2014.

[19] L. Zhuang and S. K. Tian, “Energy-saving virtual network embedding algorithm,” J. Electr. Inf. Technol., vol. 41, no. 12, pp. 3029–3035, 2019.

[20] H. Li, C. Wang, M. Li, and Y. Liu, “Virtual network embedding based on modified genetic algorithm,” Peer-to-Peer Netw. Appl., vol. 12, no. 2, pp. 481–492, Mar. 2019.

[21] X. Liu, Z. Zhang, X. Li, and S. Su, “Optimal virtual network embedding based on artificial bee colony,” EURASIP J. Wireless Commun. Netw., vol. 2016, no. 1, p. 273, Dec. 2016.

[22] P. Zhang, H. Yao, C. Fang, and Y. Liu, “Multi-objective enhanced particle swarm optimization in virtual network embedding,” EURASIP J. Wireless Commun. Netw., vol. 2016, no. 1, p. 167, Dec. 2016.
Z. Xu et al.: Energy-Driven VNE Algorithm Based on Enhanced BFO

[23] W. Cao, H. Wang, and L. Liu, “An ant colony optimization algorithm for virtual network embedding,” in Proc. ICA3PP, Dalian, China, 2014, pp. 299–309.

[24] H.-K. Zheng, J.-J. Li, Y.-J. Gong, W.-N. Chen, Z. Yu, Z.-H. Zhan, and Y. Lin, “Link mapping-oriented ant colony system for virtual network embedding,” in Proc. CEC, San Sebastian, Spain, Jun. 2017, pp. 1223–1230.

[25] F. Bianchi and F. L. Presti, “A Markov reward model based greedy heuristic for the virtual network embedding problem,” in Proc. IEE MASCOTS, London, U.K., Sep. 2016, pp. 373–378.

[26] X. Cheng, S. Su, Z. Zhang, K. Shuang, F. Yang, Y. Luo, and J. Wang, “Virtual network embedding through topology awareness and optimization,” Comput. Netw., vol. 56, no. 6, pp. 1797–1813, Apr. 2012.

[27] L. Gong, Y. Wen, and Z. Zhu, “Toward profit-seeking virtual network embedding algorithm via global resource capacity,” presented at the IEEE INFOCOM, Toronto, ON, Canada, 2014.

[28] M. He, L. Zhuang, S. Tian, G. Wang, and K. Zhang, “Multi-objective virtual network embedding algorithm based on Q-learning and curiosity-driven,” EURASIP J. Wireless Commun. Netw., vol. 2018, no. 1, pp. 1472–1687, Dec. 2018.

[29] F. Zhu and H. Wang, “A modified ACO algorithm for virtual network embedding based on graph decomposition,” Comput. Commun., vol. 80, pp. 1–15, Apr. 2016.

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

[31] K. Zhu, Z. Yao, N. He, D. Li, and L. Zhang, “Toward full virtualization of the network topology,” IEEE Syst. J., vol. 13, no. 2, pp. 1640–1649, Jun. 2019.

[32] K. M. Passino, Bacterial Foraging Optimization, vol. 1. Hershey, PA, USA: IGI Global, 2010, pp. 1–16.

[33] S. Das and A. Biswas, “Bacterial foraging optimization algorithm: Theoretical foundations, analysis, and applications,” Found. Comput. Intell., vol. 203, pp. 23–55, 2009.

[34] G. Wang, J. Guo, Y. Chen, Y. Li, and Q. Xu, “A PSO and BFO-based learning strategy applied to faster R-CNN for object detection in autonomous driving,” IEEE Access, vol. 7, pp. 18840–18859, 2019.

[35] X. Chen, C. Li, and Y. Jiang, “A feedback control approach for energy efficient virtual network embedding,” Comput. Commun., vol. 80, pp. 16–32, Apr. 2016.

LEI ZHUANG was born in Rizhao, Shandong, China, in 1963. She received the B.S. degree in computer science from Zhengzhou University, Zhengzhou, China, in 1986, and the M.S. degree in computer software from Huanghai University, Zhengzhou, in 1988, and the Ph.D. degree in computer software and theory from PLA Information Engineering University, Zhengzhou, in 2004. She is currently a Professor and a Ph.D. Supervisor with Zhengzhou University. Her main research interests include next generation internet, network virtualization, and machine learning.

LEI ZHUANG was born in Rizhao, Shandong, China, in 1963. She received the B.S. degree in computer science from Zhengzhou University, Zhengzhou, China, in 1986, the M.S. degree in computer software from Huanghai University, Zhengzhou, in 1988, and the Ph.D. degree in computer software and theory from PLA Information Engineering University, Zhengzhou, in 2004. She is currently a Professor and a Ph.D. Supervisor with Zhengzhou University. Her main research interests include future network architecture, network virtualization, and model checking.

SHUAIKUI TIAN was born in Shangqiu, Henan, China, in 1992. He received the B.S. degree in electronic and information engineering from the Anyang Institute of Technology, Anyang, China, in 2017. He is currently pursuing the M.S. degree in computer science and technology with Zhengzhou University. His main research interests include next generation internet, network virtualization, and machine learning.

MENGYANG HE was born in Nanyang, Henan, China, in 1994. She received the B.S. degree in embedding systems from Zhengzhou University, Zhengzhou, China, in 2015. She has been taking successive Postgraduate and Doctoral Programs of study for the Ph.D. degree, since 2015. She is currently pursuing the Ph.D. degree in software engineering with Zhengzhou University. Her main research interests include next generation internet, network virtualization, and machine learning.

SUJIN YANG was born in Shangqiu, Henan, China, in 1990. She received the B.S. degree in software development and the master’s degree in computer software and theory from Zhengzhou University, Zhengzhou, China, in 2012 and 2016, respectively, where she is currently pursuing the Ph.D. degree in software engineering. Her main research interests include next generation internet, network virtualization, and machine learning.

YU SONG was born in Nanyang, Henan, China, in 1969. He received the B.S. degree in computer science from Zhengzhou University, Zhengzhou, China, in 1994, and the M.S. degree in applied mathematics from the Huazhong University of Science and Technology, Wuhan, China, in 2001. He is currently an Associate Professor and a Ph.D. Supervisor with Zhengzhou University. His main research interests include artificial intelligence and machine learning.

LING MA was born in Pingdingshan, Henan, China, in 1963. He received the B.S. degree in mathematics from Zhengzhou University, Zhengzhou, China, in 1984, the M.S. degree in applied mathematics from the Hebei University of Technology, Tianjin, China, in 1987, and the Ph.D. degree in manufacturing engineering from Beihang University, Beijing, China, in 1997. He is currently a Professor and a Ph.D. Supervisor with Zhengzhou University. His main research interests include pattern recognition, image processing, and machine vision.