Zhang *et al. EURASIP Journal on Wireless Communications and Networking* (2016) 2016:167
DOI 10.1186/s13638-016-0669-2

EURASIP Journal on Wireless Communications and Networking

RESEARCH **Open Access**

Multi-objective enhanced particle swarm optimization in virtual network embedding

Peiying Zhang[1,2,3], Haipeng Yao[1*], Chao Fang[4,2] and Yunjie Liu[1]

Abstract
In network virtualization, one of its core challenges lies in how to map the virtual networks (VNs) to the shared substrate network (SN) that is managed by an infrastructure provider, termed as the virtual network embedding problem. Prior studies on this problem only consider one objective, e.g., maximizing the revenues by mapping more VNs or minimizing the energy cost. In this paper, we addressed the virtual network embedding problem with these two objectives. We leverage niche particle swarm optimization technique to design a meta-heuristic algorithm to solve this problem. Extensive simulations illustrate that the efficiency of our proposed algorithm is better than the state-of-the-art algorithms in terms of both revenue and energy cost.

Keywords: Network virtualization, Particle swarm optimization, Virtual network embedding

1 Introduction

Up to now, network virtualization has been put forward as a fundamental attribute of diversification for the future Internet by decoupling the service providers (SPs) from the roles of infrastructure providers (InPs). SPs rent some resources of the substrate network (SN) from InPs and provided services to the end users. Embedding a sequence of virtual networks (VNs) to the shared SN is defined as the VN embedding problem, which has been extensively studied [1–12].

In these previous works, the main goal is to design more efficient approaches to maximize the revenue through accommodating more virtual network requests (VNRs). Recently, in [13, 14], they proposed to minimize the energy consumption for the SPs by leveraging meta-heuristics, i.e., particle swarm optimization. Nevertheless, the particle swarm optimization (PSO) model has a high opportunity to converge to local optima. To further optimize and improve the VN embedding issues, in our works, we put forward an approach of niche particle swarm optimization algorithm (MO-NPSO) based on the multi-objectives. The originality of this algorithm is as follows. We first apply a discrete approach, which encodes the virtual node mapping solution as the position of a particle to leverage PSO. We then apply an aggregation strategy for the fitness function to transform the multi-objective problem to a uni-objective problem. To further improve the performance of this algorithm, we leverage the niche technique for the enhanced algorithm. By leveraging Niche PSO in our context, we can avoid local optima and achieve a more revenue and energy-efficient VN embedding solution in the evolution process, which is the critical contribution of this thesis. Through extensive simulations, the performance of our algorithm is better than the state-of-the-art algorithm in terms of revenue and energy cost.

The main contributions of this paper are as follows. (i) Unlike most of the prior studies, we address the virtual network embedding problem with these two objectives. (ii) We leverage niche particle swarm optimization technique to design a meta-heuristic algorithm to solve this problem. The niche technology can avoid PSO from local optima. (iii) We perform extensive simulations to prove that our proposed algorithm is better than the state-of-the-art algorithms in terms of both revenue and energy cost.

The roadmap of this thesis is 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 give details of our proposed meta-heuristic algorithm-based VN embedding which known as MO-NPSO. In

*Correspondence: yaohaipeng@bupt.edu.cn
[1]State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Xitucheng Road 10, Haidian District 100876, Beijing, People's Republic of China
Full list of author information is available at the end of the article

Section 5, we compare our MO-NPSO algorithm with other state-of-the-art algorithms. Finally, conclusions are drawn in the last section.

2 Related works

The virtual network embedding (VNE) problem, which is proved to be NP hard problem, is very hot and has been extensively studied. The primary goal earlier is to increase the long-term average revenues with certain substrate network resources. Yu et al. [1] assumed that supporting path splitting and reconfiguration can increase the opportunity of acceptance ratio of VNR. Chowdhury et al. [2] translated the VNE problem into the mixed-integer programming (MIP) problems and took advantage of classical integer programming solution to address the issue. However, this approach incurs a very high time complexity and also suffers from poor performance. Cheng et al. [10] leveraged Markov random walk theory and proposed NodeRank, similar to PageRank which serves as the core algorithm of Google's web searching. The benefit of this algorithm lies in that it incorporates the topology attribute into the VN embedding algorithm.

Another line to solve this NP hard problem is leveraging meta-heuristic. For example, Cheng and Zhang [11, 15] proposed to leverage the particle swarm optimization technique to achieve a better and better solution by iteratively searching. The difference between these two studies lies in that they did not consider the energy. Recently, due to the importance of the energy issue, energy-aware VN embedding problem also attracts several researchers. Su et al. [13, 14] and Botero et al. [16] proposed two different solutions for energy-efficient VN embedding algorithm. The former focused on the heuristics while the latter focused on the exact algorithm. However, for the heuristic algorithm, it is easy while the quality of the solution is not good; for the exact algorithm, it can achieve the optimal solution while incurring a very high time complexity. We bridge this gap in this paper and leverage the Niche PSO to address these the limitations of these two branches of algorithms.

3 The description of network model and performance metrics

3.1 The introduction of network model

We model a substrate network (SN) to a weighted graph which denoted as $G_s = (N_s, L_s)$, in which N_s stands for the collection of substrate nodes and L_s stands for the collection of substrate links.

With regard to the attributes on the nodes and links, following most of the previous studies [1, 10–13], we take the nodes' CPU capacity and location and the links' bandwidth capacity into considerations. Similar to previous studies, we model a VN to a weighted graph represented by $G_v = (N_v, L_v)$, in which N_v stands for the collection of virtual nodes and L_v stands for the collection of virtual links.

In Fig. 1, virtual network is presented on the left of diagram which is marked by (a), and substrate network is presented on the right of diagram which marked by (b). For estimating the energy consumption of the SN, the power state (with a value of on or off) of the substrate nodes is also considered. As shown in Fig. 1b, we draw such substrate nodes in the on state in transparent and such nodes in the off state in gray.

A VNR can be modeled as VNR(G_v, T_a, T_d), in which T_a stands for the arrival time and T_d stands for the duration of the VN request hosting in the SN. The VNE problem can be described as a process which maps a VN to a SN, meanwhile satisfying the resource constraints of node and link. Naturally, the mapping process can be broken into two sub-phrases: one is the node mapping phrase denoted by M_n and the other is the link mapping phrase denoted by M_l. The part of (b) in Fig. 1 gives a VNE solution for the VNR denoted by the part of Fig. 1a. From Fig. 1, we can easily know that mapping solutions.

3.2 The performance metrics

From the InP's prospective, there are two goals: (i) maximizing the revenue for serving the VN requests and (ii) minimizing the corresponding energy cost.

3.2.1 Revenue

Similar to most prior work, we take the CPU capacity revenue and bandwidth capacity of VNR as its revenue:

$$R(G_v) = \left(\sum_{n_v \in N_v} \text{CPU}(n_v) + \sum_{l_v \in L_v} \text{BW}(l_v) \right) \cdot T_d. \quad (1)$$

For embedding a sequence of VNRs, we can calculate the long-term average revenue by using the following formula:

$$\lim_{T \to \infty} \frac{\sum_{i=1}^{N} R^i(G_v)}{T}, \quad (2)$$

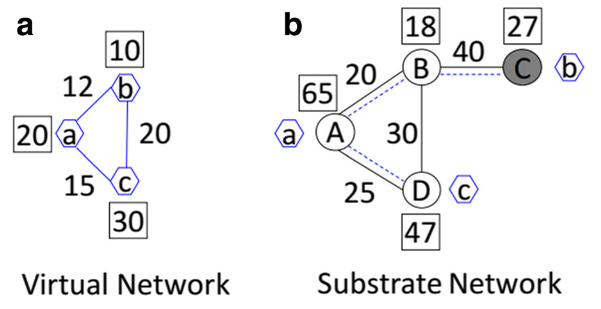

Fig. 1 Illustration of VN embedding. **a** Virtual network. **b** Substrate network

where N stands for the number of accepted VNRs in time T and $R^i(G_v)$ stands for the derived revenue for accepting the ith VNR. We quantify the resource cost for mapping G_v, denoted by $C(G_v)$ in the following ways. $C(G_v)$ consists of two parts: the first part is the total consumed CPU resource, denoted by $C(N_v)$, and the total consumed bandwidth resource, denoted by $C(L_v)$.

3.2.2 Energy cost

As reported in previous works [13, 14], the energy cost for mapping the ith VNR can be defined as:

$$E(G_v) = \left(P_l \sum_{u \in N_v} \mathrm{CPU}(u) + (N)P_b \right) T_d, \tag{3}$$

where P_l denotes a linear parameter, P_b represents the baseline energy cost, and N stands for how many substrate nodes have been powered up. Through the equation above, it is of critical importance to consolidate the VNs into substrate nodes as little as possible. To compute the energy cost for embedding a sequence of VN requests, we quantify the long-term average energy cost as follows:

$$\lim_{T\to\infty} \frac{\sum_{i=1}^{G} E^i(G_v)}{T}, \tag{4}$$

where G stands for the number of accepted VNRs in time T and $E^i(G_v)$ stands for ith VNR's energy cost. In our approach, we consider not only its long-term average revenue but also its long-term average energy cost and take both as our performance metrics.

4 Proposed solution

In particular, compared to other methods, the algorithm of PSO can produce better optimal solution with less computational time but stable convergence [17]. Furthermore, it is easy to use fewer parameters to tune. The originality of our work is to leverage these PSO benefits to construct a revenue and energy-aware VNE solution. Before that, we will give a brief introduction of particle swarm optimization basics.

4.1 Particle swarm optimization basics

In 1995, particle swarm optimization technology is first proposed by Kennedy and Eberhart, which is an emerging approach based on population optimization [18]. The basic idea of PSO borrows from the birds' flocking behavior in the process of their hunting food. In the algorithm of PSO, the problem of optimization can correspond to the process of food hunting, and the solution of optimal mapping correspond to the food's position. Each solution, which is termed as a particle in PSO, corresponds to a bird. We denote the position of ith particle by $X_i = (x_i^1, x_i^2, \ldots, x_i^D)$, in which D denotes the dimensionality. Each particle has a position. The quality of its position is better or worse, which can be determined by a fitness function. The ith particle interacts with other particles in a certain manner and adjusts its forward direction using the velocity $V_i = (v_i^1, v_i^2, \ldots, v_i^D)$ iteratively. The ith particle's velocity is determined by three factors: its current position (X_i), its best obtained experience (pBest$_i$), and the best obtained group experience (gBest). During the process of the iteration, their velocity and positions are updated by the following formula:

$$\begin{aligned} v_i^d &= P_i \cdot v_i^d + P_c \cdot r_1^d \left(\mathrm{pBest}_i^d - x_i^d \right) + \\ &\quad P_s \cdot r_2^d (\mathrm{gBest}^d - x_i^d)), \end{aligned} \tag{5}$$

$$x_i^d = x_i^d + v_i^d, \tag{6}$$

where P_i stands for the inertia weight, P_c stands for the cognition weight, P_s stands for the social weight, and r_1^d and r_2^d stand for the two random variables which are uniformly distributed in [0, 1]. In such an iterative way, the optional solution may be finally found.

It seems that we can leverage PSO to solve the VNE problem. However, for the multi-objective optimization, there still exist the following technical challenges.

1. Basic PSO only deals with the problems in continuous space. However, the VN embedding problem is discretized.

2. Basic PSO only deals with single objective optimization problem. However, this thesis can solve the multi-objective VNE problems, i.e., minimizing the resource cost and the energy consumption at the same time.

3. Basic PSO is easy to be trapped into local optima, which significantly affects the performance of our algorithm.

In the following three subsections, we will address these challenges one by one.

4.2 Discrete PSO for VNE problems

The basic PSO cannot be directly applicable to the VNE problems, since the space of VNE problem is discrete. Through analyzing the characteristics of our problem, we have put forward a discrete approach for the node mapping stage in [11, 14]. The main idea of this approach is as follows.

***Node mapping*:** We marked the virtual nodes and substrate nodes; and then, we take the position of a particle denoted by $X_i = (x_i^1, x_i^2, \ldots, x_i^D)$ as the solution of node mapping. The ith position of X_i can be assumed as the solution of the virtual node v_i mapping. That is, in this node mapping solution, the substrate node marked x_i^1 is the mapping result of the first virtual node, the substrate node marked x_i^2 is the mapping result of the second virtual node, and the substrate node marked x_i^D is the mapping result of the Dth virtual node.

Link mapping: We can also perform the link mapping in [11, 14] to compute $C(G_\mathrm{v})$ and $E(G_\mathrm{v})$. Then we take advantage of the fitness function which denoted by $C(G_\mathrm{v})$ and $E(G_\mathrm{v})$ for evaluating the quality of a particle. We also redefine the velocity and three operations for discrete PSO, including *subtraction* ($\ominus$), *addition* ($\oplus$), and *multiplication* ($\otimes$). Please see details in [11, 14].

Iteration process: After that, we renew the particle's velocity and its position with the help of the following equations.

$$\begin{aligned} v_i^d &= P_\mathrm{i} v_i^d \oplus P_\mathrm{c}(\mathrm{pBest}_i^d \ominus x_i^d) \oplus \\ &\quad P_\mathrm{s}(gBest^d \ominus x_i^d)), \quad (7) \\ x_i^d &= x_i^d \otimes v_i^d, \quad (8) \end{aligned}$$

where P_i stands for the inertia weight, P_c stands for the cognition weight, and P_s stands for the social weight. Generally, P_i, P_c, and P_s meet the following constraints: $0 < P_\mathrm{i} \le P_\mathrm{c} \le P_\mathrm{s} < 1$ and $P_\mathrm{i} + P_\mathrm{c} + P_\mathrm{s} = 1$.

4.3 Aggregation strategy for fitness function

Basic PSO only deals with single objective optimization problem. The main reason lies in that, in basic PSO, the fitness function *f*, which is used to evaluate the quality of a particle, is usually a simple uni-objective function. To solve the multi-objective optimization problem, the most common approach is aggregation. In this approach, the core idea is that all the objectives are summed to a combination. The benefit of the aggregation is that the multi-objective optimization problem can be easily transformed into uni-objective optimization problem, which helps reduce the complexity of this problem. However, this approach cannot be directly applied in our context, since $C(G_\mathrm{v})$ and $E(G_\mathrm{v})$ denotes different performance metrics and they cannot be summed directly together. Towards this end, we first normalize these two metrics as follows.

$$C(G_\mathrm{v})_\mathrm{n} = \frac{C(G_\mathrm{v}) - C(G_\mathrm{v})_\mathrm{min}}{C(G_\mathrm{v})_\mathrm{max} - C(G_\mathrm{v})_\mathrm{min}}, \quad (9)$$

$$E(G_\mathrm{v})_\mathrm{n} = \frac{E(G_\mathrm{v}) - E(G_\mathrm{v})_\mathrm{min}}{E(G_\mathrm{v})_\mathrm{max} - E(G_\mathrm{v})_\mathrm{min}}. \quad (10)$$

Based on Eqs. 9 and 10, we then define our goal as follows:

Algorithm 1 Niche PSO Algorithm

```
1: Initial $M$ particles as main particles.
2: Use basic PSO (as shown in Algorithm 2) to train $M$ particles, generally in $N$ iterations.
3: Those main particles whose degree of change of fitness is smaller than $S$ will be regarded as the center of a niche. And the radius is the distance between the center and another closest solution.
4: For each niche
5:     Use basic PSO with one iteration as shown in Algorithm 2.
6:     Update radius of each niche.
7: Allow a niche to absorb particles that moved into and niches merge together.
8: If the end criteria is met, go to step 9; otherwise, go to step 4.
9: Output the VN embedding solution and stop.
```

Minimize $f(G_\mathrm{v}) = \alpha C(G_\mathrm{v})_\mathrm{n} + (1-\alpha)E(G_\mathrm{v})_\mathrm{n}$.

Here, $\alpha \in [0,1]$. Note that, if this solution is infeasible, $C(G_\mathrm{v})_\mathrm{n}$, $E(G_\mathrm{v})_\mathrm{n}$ and $f(G_\mathrm{v})$ will be set to $+\infty$ directly.

4.4 Niche PSO

In this subsection, we will introduce how to use niche PSO into our virtual network embedding problem. Firstly, we will discuss niche technology. As well as we know, basic PSO is suffering from local optimum. To avoid this problem to some extent, Brits et al. proposed niche PSO [19]. Generally speaking, niche PSO divides the whole searching space into several small subspaces. In each subspace, we use basic PSO algorithm concurrently and independently. Besides, the niche includes absorption and merge strategy. In virtual network embedding problems, some definitions need to be discussed firstly.

The center of a niche is the best solution within that niche.

The radius R_i of a niche I is defined as:

$$R_\mathrm{i} = \max\{||X_{ib} - X_{is}||\}, \quad (11)$$

where X_{ib} is the best solution in ith niche, and X_{is} is another common solution where $b \neq s$. $||X_{ib} - X_{is}||$ denotes the Euclidean distance between the two solutions which should be calculated as $\sqrt{\sum_{k=1}^{n} (X_{ib_k} - X_{is_k})^2}$. Here, X_{ib_k} is the kth dimension of X_{ib}. The same rule holds for X_{is_k}.

Absorption of a particle into a niche is performed when:

$$||X_{js} - X_{ib}|| \le R_i, \quad (12)$$

where X_{ib} is the best solution in niche i, X_{js} is another solution which comes from niche j, and R_i is the radius of niche i, which means when a particle move into the space of another niche, it will belong to the new niche.

Merging niche is performed when the following condition is satisfied:

$$||X_{ib} - X_{jb}|| \leq (R_i + R_j), \tag{13}$$

where X_{ib} and X_{jb} are the best solutions of niche i and j, respectively, and $||X_{ib} - X_{jb}||$ is the Euclidean distance between the two solutions. Here, R_i and R_j are radius of niche i and j, which means when two niche get close enough, they will become one niche.

4.5 Description of niche PSO

In the niche PSO algorithm (Algorithm 1), the inputs are the SN, G_s and a VN request G_v, and the output is the corresponding revenue- and energy-aware VN embedding solution. We first initialize M particles as main particles. Then, we use basic PSO to train main particles. We hope to construct initial niches in this step. main particles, whose degree of change of fitness is smaller than S will be regarded as the center of a niche. And the radius is the distance between the center and another closest solution. Then in each niche, we use basic PSO concurrently and independently with on iteration to train each particle. We update the fitness and radius of each niche and allow a niche to absorb particles into and different niches merge together. If the end criteria is met, output the VN embedding solution and stop. The details of this algorithm are presented in Algorithm 1 and Fig. 2. The Algorithm 2 gives the details of the basic PSO algorithm.

5 Performance evaluation

In this section, we compare our proposed algorithm known as MO-NPSO to the prior algorithms. In Subsection 5.1, we describe our simulation settings and present our evaluation results in Subsection 5.2.

5.1 Evaluation settings

***Network topology*:** Similar to most of previous studies [1, 10–13, 15], we use the GT-ITM tool to generate the

Algorithm 2 The steps of basic PSO

```
1: Initialize position and velocity for these $M$ particles.
2: Calculate pBest and gBest.
3: For each particle
4:     Update positions and velocity of these particles according to Eqs. 5 and 6.
5:     Update fitness of each particles and update the pBest and gBest.
6: If the end criteria is met, go to step 7; otherwise, go to step 4.
7: Output the VN embedding solution and stop.
```

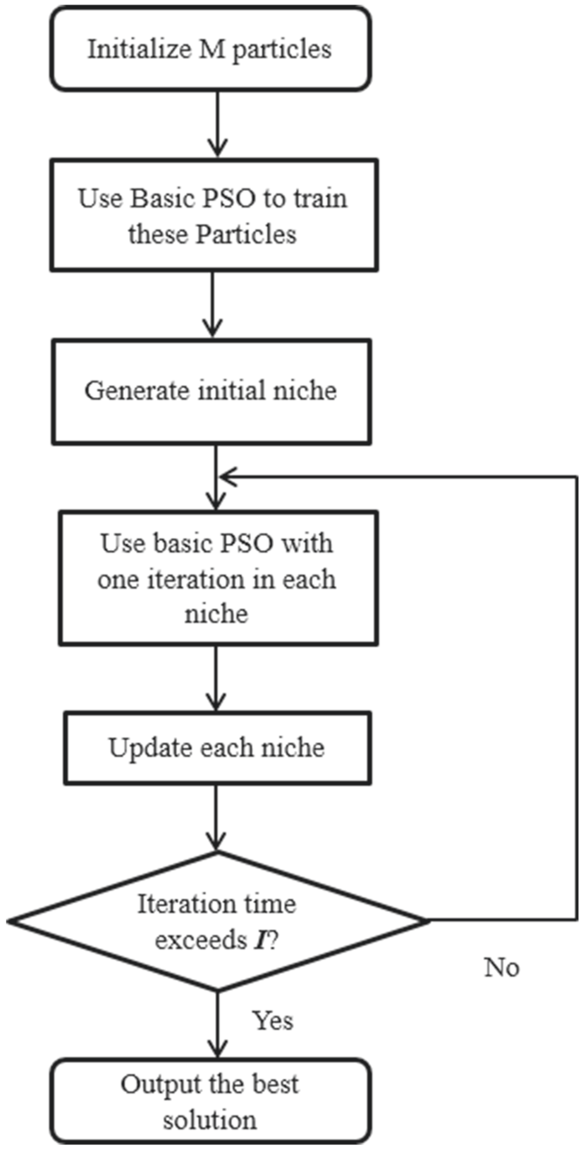
Initialize M particles
Use Basic PSO to train these Particles
Generate initial niche
Use basic PSO with one iteration in each niche
Update each niche
Iteration time exceeds I?
No
Yes
Output the best solution

Fig. 2 Illustration of MO-NPSO

topologies of substrate network and virtual network. Similar to [12, 13], the SN topology contains 50 nodes and we initialize each of them to *inactive* state. 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 the substrate nodes, it have a probability of 0.5 to connect each other. The number of nodes in a VNR has the uniform distribution of $U[2, 10]$. The average probability of link connectivity is also set to 0.5. We follow the literature [12], and each of virtual node's CPU capacity requirement obeys to the uniform distribution of $U[0, 20]$, and each of virtual node's bandwidth capacity requirement obeys the uniform distribution of $U[0, 50]$. The process of VNR's

Table 1 Compared algorithms

Notation	Algorithm description
MO-NPSO	The approach of multi-objective and meta-heuristic VNE which based on particle swarm optimization, aiming at both maximizing the revenues and minimizing the energy consumption. It adopts the adaptive weighted strategy.
EA-PSO	An energy-aware VN embedding algorithm proposed in our previous work [14]. It is also based on particle swarm optimization.
D-ViNE-SP	The single-objective VN embedding algorithm proposed in [12], aiming at maximizing the revenues by optimizing the resource cost. Take the shortest path algorithm to determine the link mapping solution after node mapping stage.

arrival can be simulated as a Poisson process, and its average arrival rate is set to 4 VNRs per 100 time units, and its residence time obeys to the exponential distribution with its average value to be 500 time units. To map a sequence of 2000 VN requests, our simulation may last for about 50,000 time units.

***Parameter settings*:** The values of P_b and P_l are assigned to 165 and 1.5 W per CPU unit, respectively, following the previous work [20, 21]. The initial value of α is set to 0.5. Following the literature [11], the value of M is set to 5, and the threshold of maximum count of iteration is set to 30. According to our experiences, the initial values of P_{i}, P_{c}, and P_{s} are set to 0.1, 0.2, and 0.7, respectively.

***Performance metrics*:** We adopt the long-term average revenue and long-average energy cost as our main performance metrics. We also calculate the resource cost of SN and the quantity of active substrate nodes, aim to explain the reason of revenue and energy cost. In addition, the running time is also another important metric in the online VN embedding.

***Comparison*:** We evaluate three algorithms as shown in Table 1. We compare our algorithm to two prior algorithms, D-ViNE-LP and EA-PSO. Note that we assume that our proposed algorithm does not support path splitting; hence, we did not compare the algorithm of MO-NPSO which supports the path splitting [12]. We also want to compare our solutions to the optimal one achieved by exact algorithm (e.g., GLPK [22]). Nevertheless, our experimental results have proved that GLPK will take several hours to calculate the global optimal solution when a VNR contains a few of nodes. It is not practical, and thus, we also exclude this comparison. The whole of our simulation experiments are conducted on the server, and its configuration is dual-core CPU with Intel 3 GHz, the memory and disk space are 2 GB and 160 GB, respectively, and the operation system is Linux 2.6.

5.2 Experimental results

We summarize our experimental results in terms of revenue comparison, energy comparison, and running time comparison.

***Revenue comparison*:** The results of average revenue comparison are shown is Fig. 3. We can observe that from the point views of long-term average revenue, our algorithm (MO-NPSO) is much better than other algorithms (The Algorithm of D-ViNE-SP and EA-PSO). From the results, we can see that the average revenues of EA-PSO and D-ViNE-SP are 4000 and 3592, respectively; the average revenue of our algorithm known as MO-NPSO is 4670. That means that our algorithm can obtain up to

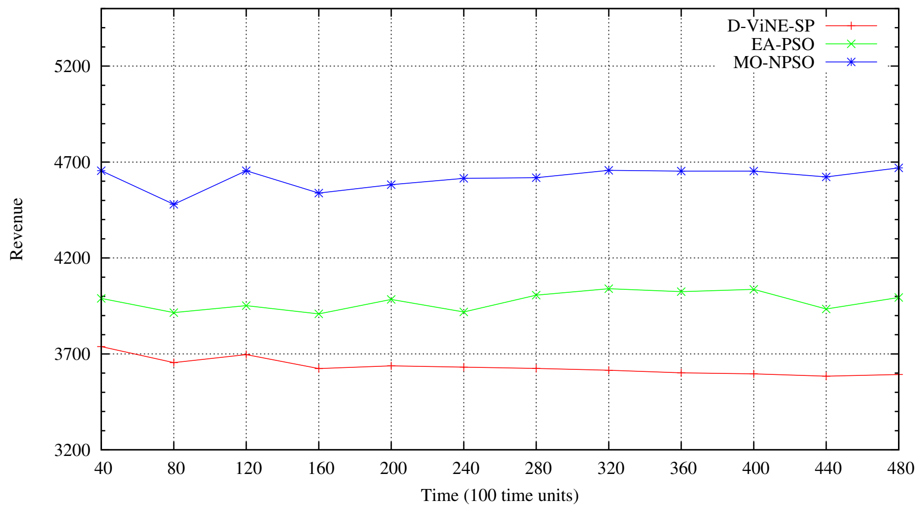

Fig. 3 The long-term average revenue over time

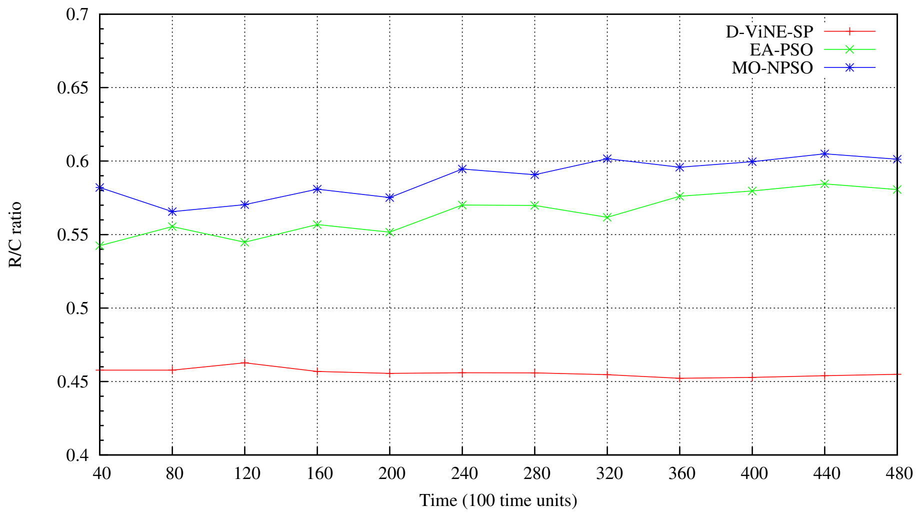

Fig. 4 The long-term average R/C ratio over time

30 % more revenue than the algorithm of D-ViNE-SP and can obtain up to 17 % more revenue than the algorithm of EA-PSO. The reason lies in that: (i) the technique of VN consolidation can improve the efficiency of substrate resources utilization, and can increase the number of VNRs. (ii) In the evolutionary process of PSO, MO-NPSO can achieve a better and better embedding solution with the help of other particles' experiences. (iii) The niche technique effectively avoids the local optima of PSO. MO-NPSO can save huge amounts of resources for the SN, accommodate more VNRs, and thus the revenues of InP will naturally increase. Comparatively speaking, the techniques of linear programming relaxation and rounding which D-ViNE-SP adopted are likely to some impractical VNE solutions, the revenues of InP will naturally decrease.

Demonstrated by Fig. 4, MO-NPSO achieves 33 % higher R/C ratio than D-ViNE-SP. Consequently, MO-NPSO generates much higher long-term average revenues than D-ViNE-SP.

***Energy comparison*:** We report the energy comparison result in Fig. 5. We can observe that while generating much more revenues, the algorithm MO-NPSO still conserves up to 17 % energy consumption than D-ViNE-SP, which ignores the energy issue and 5 % than EA-PSO, by avoiding powering up inactive substrate nodes. From Fig. 6, we can also note that MO-NPSO uses less number of substrate nodes than D-ViNE-SP. This helps demonstrate why MO-NPSO achieves more energy-efficient VN embedding solution.

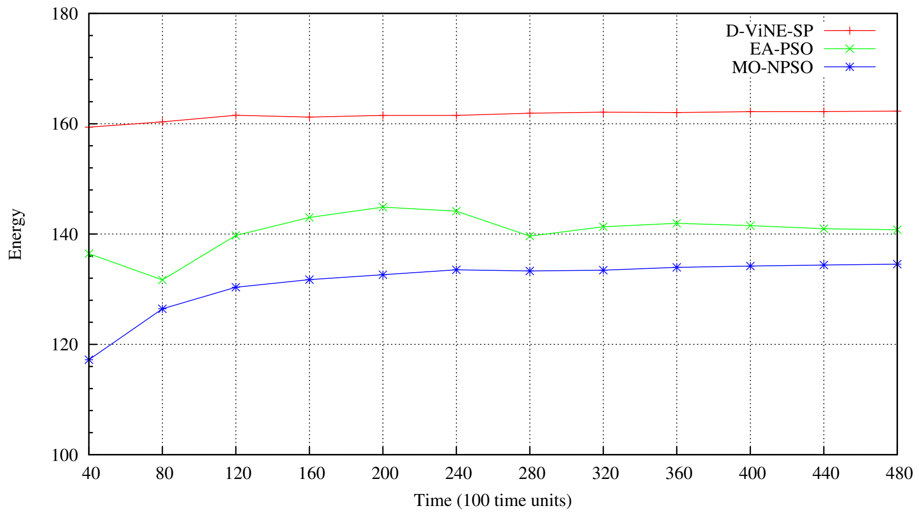

Fig. 5 The long-term average energy cost over time

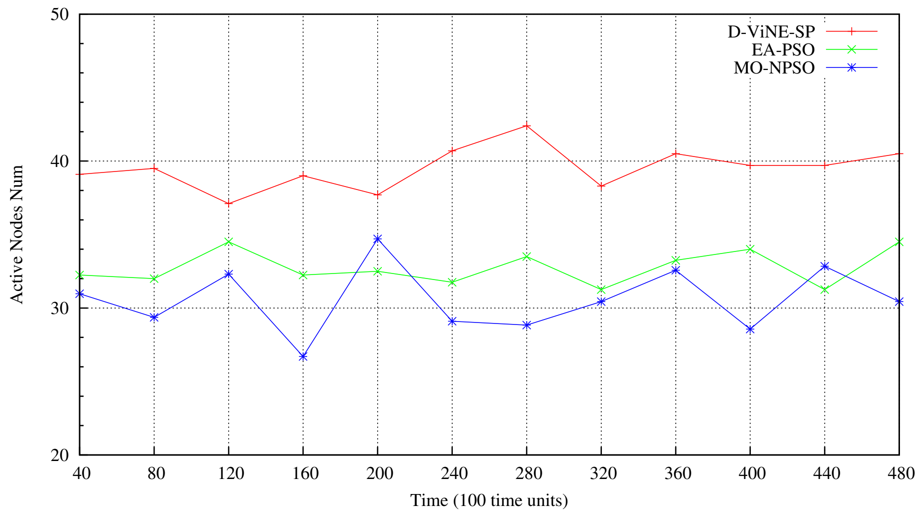

Fig. 6 The number of active substrate nodes over time

***Running time comparison*:** The average running time of compared algorithms for mapping a VNR is shown in Table 2. From this table, we see that, the time that D-ViNE-SP algorithm consumes is more than EA-PSO algorithm and MO-NPSO algorithm due to its solving linear programming [12]. The running time of MO-NPSO algorithm is a little more than EA-PSO algorithm, while it is worth for the revenue gain and energy saving.

6 Conclusions

Network virtualization is a promising technique with the purpose of overcoming the ossification of the Internet by means of supporting multiple VNs to cohabit on a shared substrate infrastructure. In the environment of network virtualization, VNE issue is proposed as one of the most important issues. In this paper, we put forward MO-NPSO algorithm to solve the multi-objective VNE problem by means of trading off the revenue and the energy cost. To be specific, we design an enhanced revenue and energy-aware VNE algorithm termed as MO-NPSO, by leveraging niche particle swarm optimization. Through extensive simulations, we prove the efficiency of our proposed algorithm.

In the future, we plan to focus on how to use path division to optimize the revenue and energy cost, and optimize energy cost while considering a variety of node types.

Table 2 The running time of compared algorithms

Algorithm	Running time
D-ViNE-SP	4 s
EA-PSO	100 ms
MO-NPSO	110 ms

Acknowledgements
This work was supported by the Shandong Provincial Natural Science Foundation, China (Grant No. ZR2014FQ018), the National Natural Science Foundation of China (Grant No. 61471056), China Jiangsu Future Internet Research Fund (Grant No. BY2013095-3-1), and BUPT Youth Research and Innovation Plan (Grant No. 2014RC0103).

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

Author details
[1]State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Xitucheng Road 10, Haidian District 100876, Beijing, People's Republic of China. [2]Beijing Advanced Innovation Center for Future Internet Technology, Beijing University of Technology, Pingleyuan 100, Chaoyang District 100124, Beijing, People's Republic of China. [3]College of Computer and Communication Engineering, China University of Petroleum (East China), Changjiang West Road 66, 266580 Qingdao, China. [4]College of Electronic Information and Control Engineering, Beijing University of Technology, Pingleyuan 100, Chaoyang District 100124, Beijing, People's Republic of China.

Received: 10 April 2016 Accepted: 5 July 2016
Published online: 13 July 2016

References
1. M Yu, Y Yi, J Rexford, M Chiang, Rethinking virtual network embedding: substrate support for path splitting and migration. Acm Sigcomm Comput. Commun. Rev. **38**(2), 17–29 (2008)
2. NMMK Chowdhury, MR Rahman, R Boutaba, in *INFOCOM*. Virtual network embedding with coordinated node and link mapping (IEEE, Rio de Janeiro, 2009), pp. 783–791
3. I Houidi, W Louati, D Zeghlache, in *IEEE International Conference on Communications*. A distributed virtual network mapping algorithm (IEEE, Beijing, 2008), pp. 5634–5640
4. J Lischka, H Karl, in *ACM Workshop on Virtualized Infrastructure Systems and Architectures*. A virtual network mapping algorithm based on subgraph isomorphism detection (ACM, New York, NY, USA, 2009), pp. 81–88
5. Y Zhu, M Ammar, in *Infocom IEEE International Conference on Computer Communications*. Algorithms for Assigning Substrate Network Resources to Virtual Network Components (IEEE, Barcelona, Spain, 2006), pp. 1–12
6. J Lu, J Turner, Efficient Mapping of Virtual Networks onto a Shared Substrate. Washington University in St Louis. **35** (2006)
7. J Fan, M Ammar, Dynamic topology configuration in service overlay networks: a study of reconfiguration policies. INFOCOM. **2**(9), 1–12 (2006)

8. C Fang, FR Yu, T Huang, J Liu, Y Liu, A survey of green information-centric networking: research issues and challenges. Commun. Surveys Tuts. **17**(3), 1455–1472 (2015)
9. C Fang, FR Yu, T Huang, J Liu, Y Liu, A survey of energy-efficient caching in information-centric networking. IEEE Commun. Mag. **52**(11), 122–129 (2014)
10. X Cheng, S Su, Z Zhang, H Wang, F Yang, Y Luo, J Wang, Virtual network embedding through topology-aware node ranking. ACM SIGCOMM Comput. Commun. Rev. **41**(2), 39–47 (2011)
11. X Cheng, S Su, Z Zhang, K Shuang, F Yang, Y Luo, J Wang, Virtual network embedding through topology awareness and optimization. Comput. Netw. **56**(6), 1797–1813 (2012)
12. NMMK Chowdhury, MR Rahman, R Boutaba, ViNEYard: Virtual network embedding algorithms with coordinated node and link mapping. IEEE/ACM Trans. Netw. **20**(1), 206–219 (2012)
13. S Su, Z Zhang, X Cheng, Y Wang, Y Luo, J Wang, in *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*. Energy-aware virtual network embedding through consolidation (IEEE, Orlando, FL, 2012), pp. 127–132
14. S Su, Z Zhang, AX Liu, X Cheng, Y Wang, X Zhao, Energy-aware virtual network embedding. IEEE Trans. Netw. **22**(5), 1607–1620 (2014)
15. Z Zhang, X Cheng, S Su, Y Wang, K Shuang, Y Luo, A unified enhanced particle swarm optimization-based virtual network embedding algorithm. Int. J. Commun. Syst. **26**(8), 1054–1073 (2013)
16. JF Botero, X Hesselbach, M Duelli, D Schlosser, A Fischer, H de Meer, Energy efficient virtual network embedding. IEEE Commun. Lett. **16**(5), 756–759 (2012)
17. P Bajpai, SN Singh, Fuzzy adaptive particle swarm optimization for bidding strategy in uniform price spot market. Power Syst. IEEE Trans. **22**(4), 2152–2160 (2007)
18. J Kennedy, RC Eberhart, et al, in *Proceedings of IEEE international conference on neural networks, volume 4*. Particle swarm optimization (Perth, Australia, 1995), pp. 1942–1948
19. R Brits, AP Engelbrecht, F Van den Bergh, in *Proceedings of the 4th Asia-Pacific conference on simulated evolution and learning, volume 2*. A niching particle swarm optimizer (Singapore, Orchid Country Club, 2002), pp. 692–696
20. LA Barroso, U Hölzle, The datacenter as a computer: an introduction to the design of warehouse-scale machines. Synth. Lectures Comput. Arch. **4**(1), 1–108 (2009)
21. G Lu, C Guo, Y Li, Z Zhou, T Yuan, H Wu, Y Xiong, R Gao, Y Zhang, in *Proceedings of the 8th USENIX conference on Networked systems design and implementation*. Serverswitch: A programmable and high performance platform for data center networks (USENIX Association, CA, USA, 2011)
22. GNU Linear Programming Kit, http://www.gnu.org/software/glpk/