Virtual Network Embedding Algorithm Based on Multi-objective Particle Swarm Optimization of Pareto Entropy

Ying Liu, Cong Wang(✉), Ying Yuan, Guo-jia Jiang, Ke-zhen Liu, and Cui-rong Wang

College of Computer and Communication Engineering, Northeastern University at Qinhuangdao, Qinhuangdao 066004, China
congwl981@163.com

Abstract. Virtual network embedding/mapping refers to the reasonable allocation of substrate network resources for users' virtual network requests, which is a key issue for virtual resource leasing in Cloud computing. Most of the existing researches only aim to maximize the revenue. As the scale of hardware network expands, the energy consumption of substrate network also needs to be paid more attention. In this paper, a multi-objective virtual network mapping algorithm based on particle swarm optimization with Pareto entropy (VNE-MOPSO) is proposed. It combines energy consumption and revenue. The algorithm controls the energy consumption of the substrate network as much as possible to achieve the goal of energy saving on the premise of ensuring a small resource cost. By introducing the Pareto entropy based multi-objective optimization model, it can calculate the difference of entropy and evaluate the evolutionary state. With this as feedback information, a dynamic adaptive particle velocity updating strategy is designed to achieve the goal of solving the approximate optimal multi-objective optimization mapping scheme. Simulation results show that the proposed algorithm has certain advantages over the typical single target mapping algorithm in cost, energy consumption and average return.

Keywords: Virtual network embedding · Multi-objective optimization · Discrete particle swarm optimization · Pareto entropy

1 Introduction

Since the 1960s, the Internet has flourished and become an important information infrastructure in modern society. On August 20, 2018, the China Internet network information center (CNNIC) released the 42nd statistical report on the development of China's Internet network [1]. The report shows that by June 2018, the number of Chinese Internet users has reached 802 million, and the Internet penetration rate has reached 57.7%. More than half of the Chinese people have been connected to the Internet. However, with the explosive growth of the number of users, as well as an increasing number of distributed applications and emerging network technologies, network "rigidity" phenomenon is becoming increasingly serious, which hinders the development of the Internet.

Q. Li et al. (Eds.): BROADNETS 2019, LNICST 303, pp. 73–85, 2019.
https://doi.org/10.1007/978-3-030-36442-7_5

Network virtualization has great prospects in the future development of the Internet. Besides, it has been actively applied in software-defined network and cloud computing environment [2]. Network virtualization takes the substrate networks as the basic entities, which enables users to take the whole network including virtual hosts and virtual links as the request scheme. Compared with the previous leasing mode that can only rent virtual machines but cannot guarantee the demand of network resource, it has better feasibility.

Virtual network embedding (VNE) problem [3] is one of the key technologies of network virtualization technology. Each virtual network request (VNR) from service providers is constrained by node resources (CPU, memory, storage) and link resources (bandwidth). The content of VNE is to allocate substrate network resources to these virtual network requests. At present, meta-heuristic algorithms have been successfully applied to a wide range of optimization problems [4] (e.g. [4–8]). Many researchers have explored the optimization model, searching strategy, algorithm acceleration, etc. They have provided important references for the follow-up researchers' works.

However, most of the studies focus on the optimization of single objective. Considering the needs to balance mapping costs, benefits, energy consumption, quality of service (QoS) and other issues in the actual demands of virtual resource leasing, we set out from the perspective of multi-objective optimization in this paper. Multi-objective optimization problem (MOP) studies the optimization of multiple objective functions in a given region. The optimization result is a Pareto optimal solution set. We model and solve the VNE problem according to reference [9].

In this paper, we combines Pareto entropy multi-objective optimization model with particle swarm optimization (PSO) to ensure the rental revenue of physical network resources, and takes the energy consumption into account at the same time. Using the target space transformation method, the external Pareto solution set is mapped to the parallel cell coordinate system, and then the population evolutionary state is evaluated according to the distribution of entropy of the approximate Pareto front end. Then, we use the feedback information of evolutionary process to design an adaptive parameter setting strategy that dynamically balances the development and utilization capabilities. The simulation results show that the proposed VNE-MOPSO algorithm effectively reduces the cost and energy consumption of virtual network mapping.

2 Problem Formulation

In this section, the VNE problem and the classification of energy consumption are presented firstly. Then, with the objective of minimizing the mapping cost and energy consumption, we establish the Integer Linear Programming (ILP) model for multi-objective optimization of VNE problem.

2.1 Virtual Network Embedding Problem

Substrate Network (SN). We mode the substrate network as a weighted undirected graph $G^S = (N^S, L^S, A_N^S, A_L^S)$, where N^S is the set of substrate nodes and L^S is the set

of substrate links. A_N^S and A_L^S denote CPU capacity of the substrate nodes and bandwidth of links, respectively.

Virtual Network (VN). Similar to the substrate network, a virtual network can be represented as $G^V = (N^V, L^V, C_N^V, C_L^V)$, where N^V and L^V denote the set of virtual nodes and virtual link respectively. Virtual nodes and edges are associated with constraints on CPU and bandwidth resources requests, denoted by C_N^V and C_L^V respectively. Each VNR can be denoted by $VNR_i = (G^V, T_i)$, where T_i denotes the duration of VN staying in the substrate network.

The VNE refers to mapping the virtual networks to the subset of the substrate networks on the premise of satisfying the nodes' and links' constraints. Generally speaking, VNE is divided into two stages: node mapping stage and link mapping stage.

2.2 Energy Consumption Modeling

The energy consumption of the substrate network is divided into two parts: the energy consumption of nodes and the energy consumption of links.

Energy Consumption of Nodes. Be similar to the earlier work in [10], we define the energy consumption of the physical nodes and links. In addition to the basic operating energy consumption, we abstract the node attributes into processor attributes. Since the energy consumption of network node is linearly related to the carried load by this node, we define the i th node energy consumption PN^i as

$$PN^i = \begin{cases} P_b + (P_m - P_b) \cdot u, & \text{if node } i \text{ is active} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where P_b is the essential baseline power, P_m denotes the total power which comes into being at the maximum capacity, and u is the utilization rate of the i th node.

Energy Consumption of Links. Because of the load-reducing engines in network virtualization environment, current network devices are insensitive to the power consumption of traffic load [11], so we regard the energy consumption of physical links as a constant [12], and we define the j th link energy consumption PL^j as

$$PL^j = \begin{cases} P_n, & \text{if link } j \text{ is powered on} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

2.3 Multi-objective VNE Problem Modeling

With the objective of minimizing mapping cost and energy consumption, the mapping optimization problem for each virtual network request can be described as:

$$\min = \begin{cases} f_1 = \alpha \sum\limits_{n^v \in N^V} cpu(n^v) + \beta \sum\limits_{l^v \in L^V} \sum\limits_{l^s \in L^S} \varphi_{l^v} \times bw(l^v) \\ f_2 = \sum\limits_{(n^i, l^j) \in P^s} (t_i \times PN^i + t_j \times PL^j) \end{cases} \tag{3}$$

where f_1 denotes the network resource expenditure for each virtual network request. $cpu(n^v)$ represents the computing capacity requirement of virtual nodes n^v, and $bw(l^v)$ represents the bandwidth capacity requirement of virtual links l^v. The parameters α and β are used to adjust the relative weights of computing resources and bandwidth resources. And $\alpha + \beta = 1$. P^S represents the set of physical nodes and links after mapping. φ_{l^v} is a binary variable used to judge whether virtual link l^v is mapped to physical link. f_2 represents the energy consumption of each virtual network request. t_i and t_j denote the length of time the node i is open and the link j is open, respectively.

Meanwhile, the mapping process must satisfy the constraints shown in Eq. (4). The first constraint is about host resource constraints. The idle resources of the current physical node need to have more resources than that requested by the virtual node to be mapped. The second constraint is about physical bandwidth constraints. Each physical link l^j occupied by each virtual link l^V, on physical path p^s, must have greater idle bandwidth than that requested by the virtual link l^V.

$$\begin{aligned} \text{s.t.} \quad & \forall n^v \in N^V, \forall n^i \in p^S, n^v \to n^i, \\ & Ccpu(n^i) - \sum_{n^v \to n^i} Ccpu(n^v) \geq Rcpu(n^v) \\ & \forall l^v \in L^V, \forall l^j \in p^S, l^v \to l^j, \\ & \min_{l^s \in l^j} Cbw(l^S) \geq Rbw(l^v) \end{aligned} \tag{4}$$

3 Virtual Network Mapping Algorithm Based on Pareto Entropy

Because the optimal solution of the above optimization model is not unique, the VNE-MOPSO algorithm proposed in this paper saves a higher quality feasible solution to the external archive (Pareto optimal solution set) whenever it is found in the iteration process. This section introduces Pareto optimal correlated definitions, Pareto entropy multi-objective optimization model, external archive updating algorithm, particle swarm optimization algorithm, adaptive parameter strategy and the overall flow of VNE-PSO algorithm.

3.1 Relevant Definitions of Pareto Optimality

Definition 1 (Pareto dominate). For any two vectors $u, v \in \Omega$, we call u dominate v (or v is dominated by u) which is denoted as $u \succ v$, if and only if $\forall i = 1, 2, \ldots, m,\ u_i \leq v_i \ \wedge \ \exists j = 1, 2, \ldots, m,\ u_j < v_j$, where m is the number of optimization objectives.

Definition 2 (Pareto optimal solution and Pareto optimal solution set). A solution $x^* \in \Omega$ is called Pareto optimal solution or non-dominant solution if and only if $\neg\exists x \in \Omega : x \succ x^*$. The set $PS = \{x^* | \neg\exists x \in \Omega : x \succ x^*\}$ of all Pareto optimal solutions is called Pareto optimal solution set.

Definition 3 (Pareto Front End). The region $PF = \{F(x^*) | x^* \in PS\}$ formed by the objective function values corresponding to all Pareto optimal solutions is called Pareto Front End or Pareto Equilibrium Surface.

3.2 Pareto Entropy Multi-objective Optimization Model and Evolutionary State

Pareto Entropy Multi-objective Optimization Model takes the difference of Pareto entropy as the basis of optimization. Firstly, we map the multi-dimensional Pareto solution stored in the external archive to the two-dimensional plane by the target space transformation method. Thus, we can obtain the parallel lattice coordinates of each Pareto solution and calculate the Pareto entropy value of the external archive approximating the Pareto front end. When the external archive is updated, the difference of entropy will be generated. Evaluating the population status according to the updating situation and using it as feedback information can better control the optimization process, taking into account the diversity and convergence of the population.

We transform the multi-dimensional Pareto solution into two-dimensional plane in parallel coordinates [9]. The integer coordinate of its mapping is the parallel cell coordinate, and the calculation formula is as follows

$$L_{k,m} = \begin{cases} \left\lceil K \frac{f_{k,m} - f_m^{\min}}{f_m^{\max} - f_{k,m}} \right\rceil, & \text{if } f_{k,m} \neq f_m^{\min} \\ 1, & \text{otherwise} \end{cases} \tag{5}$$

where $\lceil x \rceil$ returns the smallest integer that not less than x; $k = 1, 2, \ldots K$, K is the number of external archive in the current iteration, which need not be specified by the user; $m = 1, 2, \ldots M$, M is the number of objectives to be optimized; $f_m^{\max}$ and $f_m^{\min}$ are the maximum and minimum values of the m th objective of the current Pareto solution set, respectively.

In the t th iteration process, the Pareto entropy [9] of the external archive approximation Pareto front end is

$$Entropy(t) = -\sum_{k=1}^{K}\sum_{m=1}^{M} \frac{Cell_{k,m}(t)}{KM} \log \frac{Cell_{k,m}(t)}{KM} \tag{6}$$

where $Cell_{k,m}(t)$ represents the number of cell coordinate components that fall on the k th row and m th column after the approximate Pareto front end is mapped to the parallel cell coordinate system.

When the number of members of external archive reaches the maximum capacity, it is necessary to evaluate the individual density of new solution and all old solutions when updating external archive again. The individual density $Density(P_i)$ [9] of any solution P_i is as follows

$$Density(P_i) = \sum_{\substack{j=1 \\ j \neq i}}^{K} \frac{1}{PCD(P_i, P_j)^2} \tag{7}$$

$$PCD(P_i, P_j) = \begin{cases} \sum_{m=1}^{M} \left| L_{i,m} - L_{j,m} \right|, & \text{if} \exists m, L_{i,m} \neq L_{j,m} \\ 0.5, & \text{if} \forall m, L_{i,m} = L_{j,m} \end{cases} \tag{8}$$

where $i, j = 1, 2, \ldots K$, K is the number of members in external archive; P_j is any other non-dominant solution that different from P_i in external archive. $PCD(P_i, P_j)$ denotes the distance of parallel cell between P_i and P_j.

With the continuously searching of new solutions by particle swarm optimization, the external archive is constantly updated. On this ground, the evolutionary state of the algorithm in each iteration is divided into three kinds:

Stagnation state: The new solution obtained by the algorithm is denied access to external archive.
Diversified state: The new solution obtained by the algorithm replaces the old solution of poor quality in external archive.
Convergence state: The Pareto front-end generated by the algorithm approximates the real Pareto front-end in the target space.

3.3 External Archive Updating Algorithms

The main feature of the second generation multi-objective evolutionary algorithm is that external archives retain elite solutions, so it is necessary to constantly update external archive to obtain high-quality Pareto optimal solution set. In the process of updating external archive, there will be five cases [9].

Case I. If the external archive is empty, the new solution will enter the external archive directly. The population is in convergent state now.
Case II. If the new solution is dominated by any old solution in the external archive, the new solution is discarded. The population is in stagnation state now.
Case III. If the new solution dominate 0–r old solutions, we firstly remove the old solutions in the external archive which are dominated by the new solution. If the external archive is not saturated at this time, the new solution is added to the external archive. The population is in convergent state now.
Case IV. If the new solution and the old solution in the external archive are mutual non-dominant solutions, and the external archive is saturated, and the individual density of the new solution is the largest one, so the new solution is discarded. The population is in stagnation state now.
Case V. If the new solution and the old solutions in the external archive are non-dominant solutions, and the external archive is saturated, but the individual density of

the new solution is not the largest one, then the new solution replaces the old solution which has the largest individual density. The population is in diversified state now.

From the above five cases, we can get the external archive updating algorithm.

Algorithms 1. External Archive Updating Algorithms

Input: 1) External archive A to be updated;
2) Maximum capacity K of external archive;
3) The new solution P obtained by evolutionary algorithm;
Output: 1) Updated external archive A' ;
2) Evolutionary state,($state$ =0,1,2. Indicates stagnation state, diversification state and convergence state respectively.);
3) The difference of entropy $\Delta Entropy$.

1: if ($A=\varnothing$){
2: $A'=\{P\}$; $state=2$; $\Delta Entropy=\log M$;
3: return A' , $state$, $\Delta Entropy$; } /* Case I */
4: if (p is dominated by a_i , $a_i \in A$) {
5: $state=0$; $\Delta Entropy=0$;
6: return A , $state$, $\Delta Entropy$; } /* Case II */
7: if (for any $a_i \in A$, a_i is dominated by p){
8: set r is the number of old solutions dominated by p ,set $|A|$ is the current number of members of A . Firstly, set $A=A/\{a_i\}$.
9: if ($r==0$) $\Delta Entropy=\log\frac{|A|+1}{|A|}$;
10: else if ($r==1$) $\Delta Entropy=\frac{2}{MK}\log M$;
11: else if ($1<r\le|A|$) $\Delta Entropy=\frac{2}{MK}\log M+\log\frac{|A|}{|A|-r+1}$; }
12: if ($|A|<K$){
13: $A'=A\cup\{P\}$; $state=2$;
14: return A' , $state$, $\Delta Entropy$; } /* Case III */
15: else if ($A|==K$) {
16: set $B=A\cup\{P\}$, assess the individual density of all members of B .
17: find the member with the largest individual density $b_{\max}$ in B .
18: if ($P==b_{\max}$) {
19: $A'=A$; $state=0$; $\Delta Entropy=0$;
20: return A' , $state$, $\Delta Entropy$; } /* Case IV */
21: else {
22: $A'=B/\{b_{\max}\}$; $state=1$; $\Delta Entropy=\frac{2}{MK}\log M$;
23: return A' , $state$, $\Delta Entropy$; } /* Case V*/
24: }

3.4 Particle Swarm Optimization

Particle Swarm Optimization Algorithm was originally developed by J. Kennedy and R. C. Eberhart. It was proposed in 1995 as a result of studies on bird predation. For VNE problem, The location vector of the particle $X_i = [x_i^1, x_i^2, \ldots, x_i^N]$ represents a mapping scheme, in which x_i^n takes a positive integer and represents the number of the physical node to which virtual node n will be mapped. The particle velocity vector $V_i = [v_i^1, v_i^2, \ldots, v_i^N]$ represents the adjustment decision of the mapping scheme. During the evolutionary process, the position and velocity of each particle are updated as follows:

$$V_{i+1} = \omega V_i + c_1(pBest_i - X_i) + c_2(gBest_i - X_i) \tag{9}$$

$$X_{i+1} = X_i + V_{i+1} \tag{10}$$

where $\omega, c_1, c_2 > 0$ represent inertia weight, learning weight and group weight. The location vector $pBest_i$ represents individual optimal solution, and location vector $gBest_i$ represents global optimal solution of the whole group.

3.5 Adaptive Parameter Strategy

In order to better control the evolutionary process, it is necessary to continuously obtain real-time feedback information from the evolutionary environment. We can flexibly control the search trend of the algorithm by adjusting the parameters ω, c_1, c_2 of the motion equation. In this paper, a dynamic adaptive parameter adjustment strategy is designed according to the population evolution state and the difference of Pareto entropy returned by the global external archive updating algorithm, as shown in Eq. (11). Besides, $Len_\omega, Len_{c_1}, Len_{c_2}$ are the interval lengths between the maximum and minimum values of ω, c_1, c_2. Referring to literature [13], the ranges of ω, c_1, c_2 are controlled in [0.4, 0.9], [0.5, 2.5], [0.5, 2.5], respectively.

$$\omega(t) = \begin{cases} \omega(t-1) & \text{stagnant state} \\ \omega(t-1) + 0.06 \bullet Len_\omega \bullet \Delta Entropy(t) & \text{diversified state} \\ \omega(t-1) - 0.1 \bullet Len_\omega \bullet \Delta Entropy(t). & \text{convergent state} \end{cases}$$

$$c_1(t) = \begin{cases} c_1(t-1) & \text{stagnant state} \\ c_1(t-1) - 0.06 \bullet Len_{c_1} \bullet \Delta Entropy(t) & \text{diversified state} \\ c_1(t-1) + 0.1 \bullet Len_{c_1} \bullet \Delta Entropy(t) & \text{convergent state} \end{cases} \tag{11}$$

$$c_2(t) = \begin{cases} c_2(t-1) & \text{stagnant state} \\ c_2(t-1) + 0.06 \bullet Len_{c_2} \bullet \Delta Entropy(t) & \text{diversified state} \\ c_2(t-1) - 0.1 \bullet Len_{c_2} \bullet \Delta Entropy(t) & \text{convergent state} \end{cases}$$

3.6 The Whole Flow of VNE-MOPSO Algorithms

For each virtual network request, the VNE-MOPSO algorithm firstly generates the initial position of particles randomly, then judges whether it is feasible for each new

location and updates the external archive, so as to obtain the difference of entropy and evolution state. Then it updates the particle's speed and position according to the adaptive parameter strategy until the end of the iteration. The final mapping scheme is selected randomly from the external archive.

In Eq. (9), we transform the velocity vector V_{i+1} into binary value by probability mapping. The specific method is to use sigmoid function to map V_{i+1} to [0, 1] intervals as probability. If the probability is greater than or equal to the decimal of a randomly generated [0, 1] intervals, the next step speed is 1, otherwise the value is 0, as shown in Eq. (12). We use Eq. (10) to update the position of the particles, and randomly select a new physical node that satisfies host resource constraints for a virtual node whose velocity component is 1.

$$V_{i+1}^{*} = \begin{cases} 1, & \text{if rand()} \leq \text{Sigmoid}(V_{i+1}); \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

The pseudo-code of VNE-MOPSO algorithm is shown in algorithm 2.

Algorithms 2. Virtual Network Embedding Based on Pareto Entropy (VNE-MOPSO)

Input: Virtual Network G_v, Physical Network G_s;
Output: Mapping solution.

1: get the node queue and link queue of real-time idle resources in G_s;
2: for each particle instance in the population, initialize position vector;
3: initialize global external archive $gArchive = \varnothing$,initialize Individual archive $pArchive = \varnothing$;
4: for (int i = 0; i < MaxItCount ; i++){
5: if (current position is feasible){
6: use the shortest path method to generate the mapping scheme and calculate the values f_1, f_2 of objective functions;
7: for each particles, call algorithm 1 to update $gArchive$, preserve the evolutionary state and the difference of entropy at present;
8: for each particles, call algorithm 1to update $pArchive$;
9: randomly select a solution from $gArchive$,and take it as a group optimal solution $gBest$
10: select the nearest solution to the group optimal solution from $pArchive$, and take it as an individual optimal solution $gBest$;
11: according to Eq. (11), calculate the values of ω, c_1, c_2 by evolutionary state and the difference of entropy
12: according to Eq. (9-12), update the velocity vector and position vector of particles; }
13: else if (current position is not feasible){
randomly adjust the particle position; }
14: if ($gArchive$ remains unchanged consecutively for 8 rounds){
algorithm terminates; }
15: if ($gArchive \neq \varnothing$){
randomly choose a solution as the mapping scheme from $gArchive$. }

4 Simulation Results

The performance of VNE-MOPSO is compared with that of VNE-UEPSO in reference [4]. All the algorithms in this paper are implemented in Java language under Windows system. The simulation module of VNE algorithm is compiled on the platform of CloudSim3.0.3 [14]. In order to ensure the diversity of topologies, the probability of connectivity, the boundary of resource demand and the number range of nodes are used as input parameters to generate random topologies.

In the experiment, the termination condition of discrete particle swarm optimization (DPSO) is that the global optimal position is not changed consecutively for 8 rounds or the total number of iterations exceeds 30 rounds. Each experiment tests 2000 virtual network requests. The algorithm maps 70 virtual network requests in the search waiting queue for the first time to ensure that the physical network resource occupancy reaches full as soon as possible. Later it searches for the first 20 requests in the queue at a time. The relative weight ratio of computing resources and bandwidth resources in Eq. (1) is set to 1:1. The initial values of motion parameters ω, c_1, c_2 are 0.85, 0.7, 2.3. The energy consumption parameters P_m, P_b, P_n are 300, 150, 150. The maximum capacity of external archive K is 5. The physical network and virtual network parameters in the experiment are shown in Table 1.

Table 1. Basic experimental parameter setting

Parameters	Substrate network	Virtual network
Number of nodes	80	2–10
Connectivity	0.2	0.4
CPU capacity	10000	250–2500
Bandwidth capacity	10000	200–1000;600–3000
Survival time		50–500

In experiment 1, we compared the change trend of physical network revenue-cost ratio and energy consumption when receiving 2000 virtual network requests under different virtual link bandwidth parameters. As can be seen from Fig. 1 and Fig. 2, the VNE-MOPSO algorithm can reduce the mapping cost and energy consumption of physical networks. When the substrate network is idle (VNR count is around 0–800), there is little difference between the two algorithms. The main reason is that the physical network resources are sufficient in the early stage, and the search ability of particle swarm optimization is strong. The optimization effect of the VNE-MOPSO algorithm proposed in this paper is not obvious. However, with the increasing of the number of virtual network requests accepted, the optimization effect of the VNE-MOPSO algorithm continues to increase, and finally saves about 3.63%, 6.88% of the mapping cost and 9.81%, 4.64% of the energy consumption, respectively.

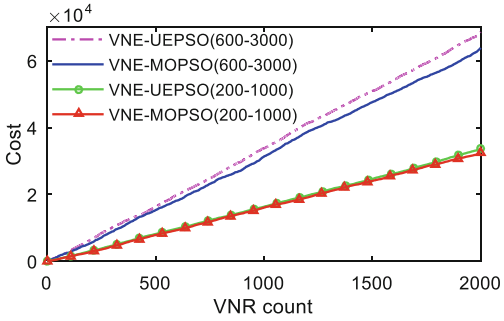

Fig. 1. Cost of substrate network under different virtual request parameters

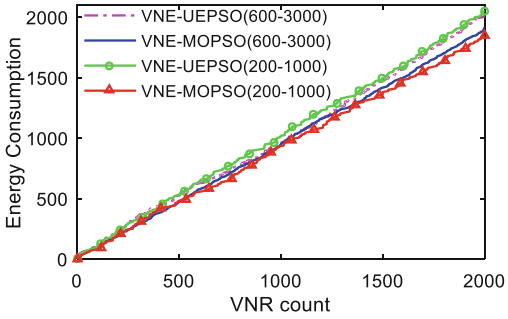

Fig. 2. Energy consumption of substrate network under different virtual request parameters

In addition, the larger the bandwidth of virtual link, the higher the corresponding mapping cost, the better the optimization effect on mapping cost the VNE-MOPSO algorithm is. Energy consumption is determined by the number of physical nodes and links that have been opened and the load of physical nodes. Energy consumption has nothing to do with the bandwidth of virtual links. Thus, the energy consumption of virtual links with bandwidth of 200–1000 is higher than that of 600–3000.

In experiment 2, we compared the trends of long-term average revenue over time when virtual link bandwidth is in 600–3000. As can be seen from Fig. 3, the VNE-MOPSO algorithm proposed in this paper has advantages over VNE-UEPSO in terms of long-term average revenue in physical networks. This is because the VNE-MOPSO algorithm takes mapping cost and energy consumption as optimization objectives, takes into account the diversity and convergence of Pareto front-end in the evolutionary process, saves physical network resources, and speeds up mapping speed, thus it can improve the long-term average revenue. In Fig. 3, the long-term average revenue is relatively high in the early stage, and then shows a downward trend. This is due to the abundant physical network resources, high mapping efficiency and the large number of virtual network requests accepted in the early stage, so the revenue is also high. However, with the increasing number of accepted virtual network requests, the process of virtual network mapping gradually reaches a stable state in the process of occupying and releasing physical resources, and the average revenue tends to be stable.

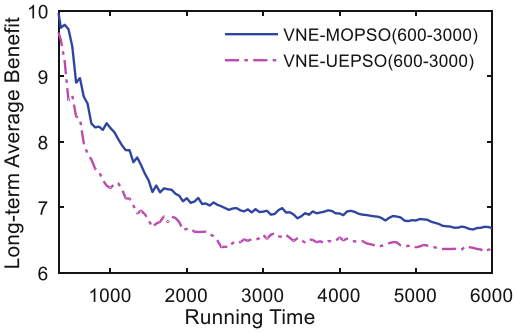

Fig. 3. The trend of long-term average benefit over time

5 Conclusions

Aiming at the VNE problem, considering the cost of mapping and energy consumption, and combining with the theory of multi-objective optimization, a multi-objective particle swarm optimization VNE algorithm based on Pareto entropy is proposed. According to the update status of the external Pareto solution set, the difference of entropy is calculated and the evolution status of the population is evaluated. Combining with the dynamic adaptive particle velocity update strategy, the mapping cost and energy consumption are reduced, and the long-term average benefit is also obtained. Experiments show that the algorithm proposed in this paper has certain advantages over other similar algorithms in terms of revenue, energy consumption and solution efficiency.

Virtual network mapping is a key problem in cloud resource allocation, and many factors need to be considered in the designing of algorithm. This paper considers multi-objective optimization from the perspective of mapping cost and energy consumption of substrate physical network. The next step is to try to optimize more objectives simultaneously and introduce virtual network quality of service as an additional parameter.

References

1. The 42nd China statistical report on Internet development. http://www.cnnic.cn/hlwfzyj/hlwxzbg/hlwtjbg/201808/t20180820_70488.htm. Accessed 20 Mar 2019
2. Fischer, A., Botero, J., Beck, M., et al.: Virtual network embedding: a survey. IEEE Commun. Surv. Tutor. **15**(4), 1888–1906 (2013)
3. Cheng, X., Zhang, Z., Sen, S., et al.: Survey of virtual network embedding problem. J. Commun. **32**(10), 143–151 (2011)
4. Zhang, Z., Cheng, X., Sen, S., et al.: A unified enhanced particle swarm optimization-based virtual network embedding algorithm. Int. J. Commun. Syst. **26**(8), 1054–1073 (2013)
5. Wang, W., Wang, B., Wang, Z., et al.: The virtual network mapping algorithm based on hybrid swarm intelligence optimization. J. Comput. Appl. **34**(4), 930–934 (2014)

6. Wang, Q.: Research on Virtual Network Mapping Algorithm Based on Particle Swarm Optimization. Northeastern University, Shen yang (2015)
7. Wang, C., Yuan, Y., Peng, S., et al.: Fair virtual network embedding algorithm with topology pre-configuration. J. Comput. Res. Dev. **54**(1), 212–220 (2017)
8. Zheng, H., Li, J., Gong, Y., et al: Link mapping-oriented ant colony system for virtual network embedding. In: IEEE Congress on Evolutionary Computation 2017, pp. 1223–1230. IEEE, Piscataway (2017)
9. Wang, H., Yen, G.G., Zhang, X.: Multiobjective particle swarm optimization based on Pareto entropy. J. Softw. **25**(5), 1025–1050 (2014)
10. Chen, X., Li, C., Chen, L., et al.: Multiple feedback control model and algorithm for energy efficient virtual network embedding. J. Softw. **28**(7), 1790–1814 (2017)
11. Chabarek, J., Sommers, J., Barford, P., et al: Power awareness in network design and routing. In: The 27th Conference on Computer Communications IEEE 2008, pp. 457–465. IEEE, Phoenix (2008)
12. Lu, G., Guo, C., Li, Y., et al: ServerSwitch: a programmable and high performance platform for data center networks. In: The 8th USENIX Conf. on Networked Systems Design and Implementation 2011, pp. 1–14. USENIX Association, Berkeley (2011)
13. Ratnaweera, A., Halgamuge, S., Watson, H.: Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans. Evol. Comput. **8**(3), 240–255 (2004)
14. Calheiros, R.N., Ranjan, R., Beloglazov, A., et al.: CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw. Pract. Exp. **41**(1), 23–50 (2011)