ARCHIVES OF ELECTRICAL ENGINEERING VOL. 69(3), pp. 595–608 (2020)

DOI 10.24425/aee.2020.133920

Heuristic energy-saving virtual network embedding algorithm based on Katz centrality

QIANG ZHU[1], QING-JUN WANG[1], MU-JUN ZANG[2], ZHEN-DONG WANG[2], CHANG XIAO[3]

[1] *School of Information and Electrical Engineering, Ludong University*
China

[2] *Jiangxi University of Science and Technology*
China

[3] *School of Economics, Ocean University of China*
Tsingtao, China
e-mail: zhqiang211@163.com

(Received: 10.10.2019, revised: 12.03.2020)

Abstract: Current networks are designed for peak loads leading to low utilization of power resources. In order to solve this problem, a heuristic energy-saving virtual network embedding algorithm based on the Katz centrality (Katz-VNE) is proposed. For solving an energy-saving virtual network embedding problem, we introduce the Katz centrality to represent the node influence. In order to minimize the energy consumption of the substrate network, the energy-saving virtual network embedding problem is formulated as an integer linear program, and the Katz-VNE is used to solve this problem. The Katz-VNE tries to embed the virtual nodes onto the substrate nodes with high Katz centrality, which is effective, and uses the shortest paths offering the best factor of bandwidths to avoid the hot nodes. The simulation results demonstrate that the long-term average energy consumption of the substrate network is reduced significantly, and the long-term revenue/cost ratio, the acceptance rate of virtual network requests, and the hibernation rate of substrate nodes as well as links are improved significantly.

Key words: energy-saving virtual network, integer linear programming, Katz centrality, network virtualization embedding, virtualization

1. Introduction

With the increase of the global greenhouse effect, how to reduce carbon emission has become the primary task of major environmental organizations and conferences. Low carbon energy

---

conservation has become a global hot topic [1]. The electricity consumption used by the Internet equipment, such as switches, routers, data centers and so on, consumes about 6% of global electricity consumption. It is expected to grow to 40% in the next 10 years. The current Internet architecture and operating models run counter to low carbon energy conservation. Important indicators such as the processing and switching capabilities of network devices are designed for peak loads of the current network. Although it can ensure the normal work of the network, 11% to 50% of the resource utilization will lead to waste of power resources inevitably. Network virtualization is an effective way to bear on this problem. While satisfying user service requirements, the network virtualization can allocate part of substrate network (SN) devices to virtual network (VN) requests and hibernate the rest of the SN devices as much as possible to achieve energy saving [2]. In network virtualization environment, the allocation of the SN resources is NP-hard called a virtual network embedding (VNE) problem [3,4].

Previous researches on the VNE problem are almost always related to efficiency [5, 6] and reliability [7–9]. The energy-saving virtual network embedding (ES-VNE) problem was proposed by Botero [10], while the time complexity of the solution is too high to adapt to large scale networks. Botero *et al.* [11] proposed an EA-VNE algorithm based on a shortest path strategy, which considering minimum hops in link allocation reduced the time complexity to a certain extent. Wang *et al.* [12] proposed energy-aware node mapping (ENM) functionally similar to the EA-VNE. The disadvantage of the algorithms based on the shortest path is that it chooses the minimum hops without considering the bandwidth usage, leading to the hot spots and reducing the acceptance rate of VNs. Sun *et al.* [13] formulated the ES-VNE problem as an integer linear program, and proposed an algorithm to solve the ES-VNE problem efficiently for cloud-based data centers. Chen *et al.* [14] proposed an ES-VNE algorithm based on hibernating substrate resources actively by analyzing dynamic characteristics and constructing VNE dictionary database. The main problem is that the time complexity is too high to adapt to large scale networks. Liao *et al.* [15] proposed an ES-VNE algorithm by using the modified open shortest path first (OSPF) protocol to determine a candidate link set in which all the links were probably closed or sleep, and remap the traffic volume of these links to other substrate paths. The rest of the substrate links can be powered off to save energy. Chen *et al.* [16] proposed an ES-VNE algorithm in which the minimization of energy consumption was solved as the minimal product of energy cost per CPU. Zhang *et al.* [17] proposed an algorithm based on particle swarm optimization with the consideration of the SN energy cost in accommodating VN requests, which is the improvement of previous researches that mainly focused on maximizing the revenue of an SN. Zheng *et al.* [18] proposed an ES-VNE algorithm based on a group search optimizer for cloud computing networks by consolidating resources in the network and data centers. To sum up, previous researches on the ES-VNE problem mainly focus on the minimization of energy consumption without considering the importance of nodes in an SN. In social network research, the measurement of nodes, called centrality, is the most direct measurement of the nodes' central degree. The node with a higher central degree is more important. We can solve the ES-VNE problem of a social network, with the help of centrality.

To reduce the energy consumption of an SN and improve an acceptance rate, a heuristic energy-saving virtual network embedding algorithm based on the Katz centrality is proposed. The Katz-VNE tries to embed virtual nodes onto substrate nodes with high Katz centrality, which is effective, and uses an avoidance factor of hot spots to avoid hot nodes.

2. Energy-saving virtual network embedding formulation

In this section, a mathematical formulation for the ES-VNE problem is provided. A network energy consumption model is proposed. ES-VNE formulation is composed by the SN and VN request modeling, VNE problem modeling and ES-VNE analysis.

2.1. Network energy consumption model

The energy consumption of an SN includes node and link energy consumption. Similar to the previous researches in literature [10] to [18], the node energy consumption is represented by CPU energy consumption, which consists of basic and load energy consumption. Link energy consumption is mainly derived from the energy consumed when the link is in operation. Literature [19] proposed a general energy consumption model, indicating that the energy consumption of SN nodes increases linearly to the full load energy consumption, as VN node load increases. According to this model, the energy consumption associated with embedding a VN request onto the SN successfully, can be formulated by:

$$E(G_i^v) = \Bigg[\sum_{n_p^s \in N^s} \alpha_{n_p^s} (\gamma_{n_p^s} - l_{n_p^s}) + \sum_{(p,q) \in E^s} \beta_{e_{pq}^s} (y_{e_{pq}^s} - x_{e_{pq}^s}) + \\ + \sum_{n_p^s \in N^s} \mu_{n_p^s}^{PP} \left(\mu'_{n_p^s} - \mu_{n_p^s} \right) cpu\,(n_p^s) \Bigg] \cdot T(G_i^v), \tag{1}$$

where $l_{n_p^s}$ and $\gamma_{n_p^s}$ are the binaries that denote the running status of the substrate node n_p^s. $\gamma_{n_p^s}$ is the status before embedding the virtual network request G_i^v and $l_{n_p^s}$ is the status after embedding the virtual network request G_i^v. If the substrate node n_p^s is in running status,

$\gamma_{n_p^s} - l_{n_p^s}$ equals 1, otherwise 0. $\alpha_{n_p^s}$ denotes the basic energy consumption of the running substrate node n_p^s per unit time. $x_{e_{pq}^s}$ and $y_{e_{pq}^s}$ are the binaries that denote the running status of the substrate link e_{pq}^s before and after embedding virtual links. If the substrate link e_{pq}^s is in the running status, $x_{e_{pq}^s}$ and $y_{e_{pq}^s}$ equal 1, otherwise 0. $\beta_{e_{pq}^s}$ denotes the basic energy consumption of the running substrate link e_{pq}^s. $\mu_{n_p^s}^{PP}$ is the unit CPU energy consumption when the substrate node n_p^s carries the virtual network node in unit time. $\mu'_{n_p^s}$ denotes the load strength of the substrate node n_p^s after embedding the virtual network request G_i^v. $\mu_{n_p^s}$ denotes the load strength of the substrate node n_p^s before embedding the virtual network request G_i^v.

2.2. SN and VN request

The SN can be described as $G^s = (N^s, E^s, A_n^s, A_e^s)$, which is an undirected graph. N^s and E^s denote the substrate node sets and substrate link sets. A_n^s denotes the available CPU computing capability $cpu\,(n^s)$ set. We sample every time unit that becomes a discrete function. A_e^s denotes each substrate link $e_{ij}^s \in E^s$. $b\,(e^s)$ denotes available bandwidth capability. P^s denotes the loop-free path set in the SN and $P^s(ij)$ denotes the loop-free path between i and j. Similar to the SN, the VN request is also described as $G^v = (N^v, E^v, A_n^v, A_e^v, T)$. T is the life cycle of the VN request. T means a period of time. When virtual network request (VNR) is accepted, the time starts until

T is reached. The VNR is terminated. Resources used by the VNR are recycled by the substrate network. The other parameter settings of the VN request are the same as that of the SN.

2.3. VNE problem modeling

The VNE problem is defined as embedding VN G^v onto $G^{s'}$.The CPU and bandwidth constraints in G^v are all needed to be satisfied. The VNE can be divided into node and link embedding.

1. Node embedding: $f_N : (N^v, C_n^v) \to (N^{s'}, A_n^s)$. According to the CPU constraints, it embeds each virtual node onto a substrate node.

2. Link embedding: $f_E : (E^v, C_e^v) \to P^{s'}$, where $P^{s'} \subset P^s$. According to link constraints, f_E embeds a virtual link onto a loop-free path in an SN.

3. Energy-saving virtual network embedding evaluation metrics

Being different from the traditional VNE algorithm, the ES-VNE algorithm needs to consider not only the acceptance rate of VN requests and average energy consumption, but also the hibernation rate of SN resources and long-term revenue/cost ratio, to reduce energy consumption of the SN as well as to improve acceptance rate as much as possible in the long run [21, 22].

1. Acceptance rate of VN requests

The acceptance rate of VN requests η_{accepted} can be formulated by

$$\eta_{\text{accepted}} = \lim_{T\to\infty} \frac{\sum\limits_{t=0}^{T} \text{VNR}_{\text{accepted}}}{\sum\limits_{t=0}^{T} \text{VNR} + \delta}, \tag{2}$$

where $\sum\limits_{t=0}^{T} \text{VNR}_{\text{accepted}}$ denotes the VN requests that has been embedded successfully between 0 and T. $\sum\limits_{t=0}^{T}$ VNR denotes the VN requests that arrive between 0 and T. δ is a positive number close to zero infinitely.

2. Average energy consumption of embedding VNs

η_{energy} is the ratio of the total energy consumed to the number of VN requests that has been accepted.

$$\eta_{\text{energy}} = \lim_{T\to\infty} \frac{\sum\limits_{t=0}^{T} E(G^v, t)}{\sum\limits_{t=0}^{T} \text{VNR}_{\text{accepted}} + \delta}, \tag{3}$$

where $\sum\limits_{t=0}^{T} E(G^v, t)$ is the total energy consumed by embedding VNs between 0 and T. $\sum\limits_{t=0}^{T} \text{VNR}_{\text{accepted}}$ denotes the VN requests that has been accepted between 0 and T.

3. Hibernation rate of substrate nodes and links

The hibernation rate of substrate nodes and links is the difference value between active nodes and links as well as total nodes and links, respectively, in the SN.

$$\eta_{N\mathrm{rate}} = 1 - \frac{|N'|}{|N| + \delta}, \tag{4}$$

$$\eta_{E\mathrm{rate}} = 1 - \frac{|E'|}{|E| + \delta}, \tag{5}$$

where $|N|$ denotes the total nodes and $|E|$ denotes the total links. $|N'|$ denotes the active nodes and $|E'|$ denotes the active links.

4. Long-term revenue/cost ratio

$\eta_{\text{revenue-cost}}$ is the ratio of the total energy consumed to the number of VN requests that has been accepted.

$$\eta_{\mathrm{revenue-cost}} = \lim_{T\to\infty} \frac{\sum\limits_{t=0}^{T} R(G^v, t)}{\sum\limits_{t=0}^{T} C(G^v, t)}, \tag{6}$$

$$R(G^v, t) = \sum_{n^v \in N^v} cpu\,(n^v) + \sum_{e^v \in E^v} b(e^v), \tag{7}$$

$$C(G^v, t) = \sum_{e^v \in E^v} \sum_{e^s \in E^s} b(e^v) + \sum_{n^v \in N^v} cpu\,(n^v), \tag{8}$$

where $\sum\limits_{t=0}^{T} R(G^v, t)$ is the total revenue of embedding VN successfully between 0 and T. $C(G^v, t)$ is the total cost between 0 and T. $\sum\limits_{e^v \in E^v} \sum\limits_{e^s \in E^s} b(e^v)$ is the total bandwidth cost of the SN.

4. Integer linear programming formulation

Objective Function

$$\min \left[\sum_{n_p^s \in N^s} \alpha_{n_p^s} (\gamma_{n_p^s} - l_{n_p^s}) + \sum_{(p,q) \in E^s} \beta_{e_{pq}^s} (y_{e_{pq}^s} - x_{e_{pq}^s}) + \right.$$
$$\left. + \sum_{n_p^s \in N^s} \mu_{n_p^s}^{PP} (\mu'_{n_p^s} - \mu_{n_p^s}) cpu\,(n_p^s) \right] \cdot T(G_i^v). \tag{9}$$

Capacity constraints

$$\forall\, i \in N^s, \quad \forall\, m \in N^v, \quad \eta_i^m cpu\,(m) \le cpu\,(i), \tag{10}$$

$$\forall\, (i, j) \in N^s, \quad \forall\, (mn) \in N^v, \quad \phi_{ij}^{mn} b\,(e_{mn}) \le b\,(e_{ij}). \tag{11}$$

Flow constraints

$$\forall\, i \in N^s, \quad \forall\, (m,n) \in E^v, \qquad \sum_{(i,j)\in E^s} \phi_{ij}^{mn} - \sum_{(j,i)\in E^s} \phi_{ji}^{mn} = \begin{cases} 1 & \text{if } \eta_i^m = 1 \\ -1 & \text{if } \eta_i^n = 1 \\ 0 & \text{otherwise} \end{cases}. \tag{12}$$

Other constraints

$$\forall\, i \in N^s, \quad \forall\, m \in N^v, \quad \eta_i^m \in 0,1\} \quad \forall\, i \in N^s, \quad \sum_{m\in N^v} \eta_i^m \leq 1, \tag{13}$$

$$\forall\, m \in N^v, \quad \sum_{i\in N^s} \eta_i^m = 1. \tag{14}$$

Remarks

1. Objective function (8) minimizes the power consumption of the SN.
2. Capacity constraints (9) and (10) indicate that the remaining resources of substrate nodes and links need to meet the needs of virtual nodes and links.
3. Flow constraints (11) mean that if the virtual nodes m and n are embedded onto the substrate nodes i and j, the virtual links (mn) are embedded to the substrate link $P \in P^s(ij)$. If i is the source node, m is embedded onto i and $\eta_i^m = 1$. The inflow of the source node is 0 and the outflow is 1. If i is the destination node, n is embedded onto i and $\eta_i^n = 1$. The inflow and the outflow of the destination node are 1 and 0, respectively. For other nodes, the inflow and outflow are both 0.
4. Constraints (12) indicate that each virtual node in the VN can only be embedded onto one substrate node. Constraints (13) indicate that nodes in the same VN cannot be embedded onto the same substrate node.

5. Heuristic energy-saving virtual network embedding algorithm based on Katz centrality

The concept of centrality originated from social network research. In network analysis, centrality is the most direct measurement of the nodes' central degree. The higher central degree one node has, the more important the node is in the network. There are many researches on measuring the social network centrality. The Katz centrality is based on random walk. On the walk path, the farther away from the node, the smaller contribution to the node under the action of the penalty factor. The random walk measurement method can solve the hot spot limitation of the shortest path measurement methods. Different from the methods based on the shortest path, which just considers the influence of the current node, the Katz centrality measurement can consider the relationship between all the network nodes and this node. With the increase of VN requests, the bottlenecks of the SN can be avoided as much as possible, which can ensure the acceptance rate of the VN under energy-saving.

Definition 1 ***Node capability factor***

The overall capacity of one node is the product of its CPU capability $cpu\,(n^s)$ and the sum of the bandwidth of the link which is connected with the node directly.

$$\upsilon(n^s) = cpu\,(n^s) \cdot \sum_{e^s \in \sigma(n^s)}^{T} e(n^s). \tag{15}$$

Definition 2 ***Node centrality correcting factor***

ES-VNE needs to prioritize nodes already in the working state when selecting SN nodes. Then, a node centrality correcting factor can improve the centrality of the nodes in the working state, which makes it more likely to be selected in the next embedding.

$$\left|\vec{C}_{\text{katz}}(n_i^s)\right|' = \left|\vec{C}_{\text{katz}}(n_i^s)\right|^{\upsilon(n_i^s)}. \tag{16}$$

Definition 3 ***Hot spots avoidance factor***

The previous works use the Dijkstra shortest path for the allocation of links, which considers the minimum hops without considering the bandwidth usage, leading to the hot spots and reducing the acceptance rate of VNs. The hot spot avoidance factor $\psi(p)$ is proposed to solve this problem.

$$\psi(p) = \frac{\text{length}(p)}{\min_{e_{ij}^s \in p} b\,(e_{ij}^s)}. \tag{17}$$

Definition 4 ***Link energy penalty factor***

The ink energy penalty factor ϑ can prevent the SN from the bottleneck link. The link overload occurs when the load factor ε of the SN link exceeds the value ρ, leading to the bottleneck link. The bottleneck link can affect the nodes and links connected with it. When the load factor ε is in a certain range, the link energy consumption has a normal value. Otherwise, the load factor ε will be amplified by the link energy penalty factor ϑ.

$$\beta_{e_{pq}^s} = \begin{cases} \beta_{e_{pq}^s} & \varepsilon \in (0, \rho] \\ \vartheta \beta_{e_{pq}^s} & \varepsilon \notin (0, \rho] \end{cases}, \tag{18}$$

$$\varepsilon = \frac{\sum\limits_{\forall e^v \perp e^s} b(e^v)}{\max b(e^s)}, \tag{19}$$

where $\max b(e^s)$ is the maximum bandwidth of the substrate link. $e^v \perp e^s$ denotes the virtual link e^v which is allocated to the substrate link e^s.

Unlike the traditional VNE algorithms based on minimum cost, an ES-VNE algorithm needs to consider both the capability of a substrate node and its centrality in the entire SN, which is influenced by nodes. Embedding nodes onto the SN nodes with high capabilities and influence, and using a hot spot avoidance factor and link energy penalty factor, allow one to avoid the hot spots in the SN as much as possible and have a positive impact on the embedding of other nodes and links in the VN. The basic workflow of the Katz-VNE is presented using a pseudocode in Algorithm1 and is described as follows:

Algorithm 1 Heuristic energy-saving virtual network embedding algorithm based on the Katz centrality

```
Input: $G^s = (N^s, E^s, A_n^s, A_e^s)$ $G^v = (N^v, E^v, C_n^v, C_e^v, T)$
Output: $f_N$ $f_E$
Initialization: $Q^v = \{G_1^v, G_2^v, ..., G_n^v\}$
If $Q^v \neq \emptyset$ then
    Take the first virtual network request in the queue $Q^v$ and put it in VN,
VN = $Q^v pop$;
for $i = 1$; $i \leq |N|$; $i = i + 1$ do
Use $|\vec{C}_{\text{katz}}(i)| = |\left((I - \alpha A^{-1}) - I\right)\vec{1}^T|$ to get the Katz centrality $|\vec{C}_{\text{katz}}(i)|$ of the virtual node.
        if $|\vec{C}_{\text{katz}}(i)| \neq |\vec{C}_{\text{katz}}(j)|$then
            Put the virtual nodes in descending order according to $|\vec{C}_{\text{katz}}(i)|$.
        else
            Put the virtual nodes with the same $|\vec{C}_{\text{katz}}(i)|$ according to
            $\upsilon(n_i^v) = cpu\,(n_i^v) \cdot \sum_{e^v \in \sigma(n_i^v)} e(n_i^v)$ in descending order.
        end if
end for
Use $|\vec{C}_{\text{katz}}(i)|$ to get the Katz centrality for each node in the substrate network and put them in descending order.
For $i = 1$; $i \leq |N|$; $i = i + 1$ do
        Allocate the virtual node to the substrate node in descending order according to
        $|\vec{C}_{\text{katz}}(i)|$ and $\upsilon(n_i^v)$.
        Fix the centrality of the node that is working
        For $i = 1$; $i \leq |E|$; $i = i + 1$ do
            Select the best path for the virtual network link in the set of the loop-free path $P^s(ij)$
            between $i$ and $j$ by using the product of $\psi(p)$ and $\beta_{e_{ij}^s}$.
        end for
        Update the energy consumption of each link by using the link energy penalty factor
end for
end if
if virtual network request lifetime = 0 then
    Release resources occupied by the virtual network and update the resource usage of the
    substrate network
end if
end
```

Step 1. Check the VN request queue. If it is empty, end the algorithm; otherwise, turn to step 2.

Step 2. Get the Katz centrality and node capability factors for each SN and VN node and put them in descending order. If different nodes have the same Katz centrality, put them in descending order according to the node capability factor.

Step 3. According to constraints (9), (12) and (13), VN nodes are sequentially embedded in descending order. If the constraint of the VN node cannot be satisfied, the VN request is rejected; otherwise, turn to step 4.

Step 4. Use a node centrality correcting factor to correct the centrality after selecting nodes.

Step 5. According to constraints (10) and (11), use a hot spot avoidance factor and the basic energy consumption of the substrate link $\beta_{e^s_{ij}}$ to get current link energy consumption, and find the best path in $P^s(ij)$. The minimum product of the path is the best path. If $P^s(ij)$ cannot satisfy the constraint of the virtual link, reject the VN request; otherwise, turn to step 6.

Step 6. After embedding the links, use the link energy penalty factor to update the energy consumption of each link.

6. Experimental design and results

6.1. Simulation environment

The graphs for the substrate network and virtual network requests are generated by using the GT-ITM topology generator tool [5], which is widely used for generating network topologies. The simulation settings for the Katz-VNE are described as follows. The SN has 50 nodes with 0.5 probability connection. Both available, a CPU and link as well as width resources with a uniform distribution of 50 to 100. VNs arrive in a Poisson process with an average rate of 4 virtual network requests per 100 time units, and each request has an exponentially distributed lifetime with an average of 1000 time units. Virtual nodes in each VN are varied from 2 to 5. Probability of virtual nodes connection is 0.5. Bandwidth requirement of virtual link with a uniform distribution of 20 to 100. Basic energy consumption of the substrate link $\beta_{e^s_{pq}}$ is 15 W. Basic energy consumption per unit of the active substrate node $\alpha_{n^s_p}$ is 150 W. When loading a virtual node, the unit energy consumption of the substrate node $\mu^{PP}_{n^s_p}$ is 1 W. The node centrality correcting factor $\upsilon(n^s_i)$ is $\sqrt{2}$ and ρ is 0.8. Before doing comparison experiments, we performed multiple verifications of the value of ρ. The experimental results show 0.8, which is the best value of ρ. We repeated the experiment 10 times to get the average.

Table 1 shows four algorithms evaluated in our simulation experiments. In EA-VNE, the energy-saving VNE algorithm is based on the shortest path. D-ViNE is a non-energy-saving VNE algorithm based on the shortest path. JLB-VNE is a joint load balancing and energy saving algorithm for virtual network embedding [19, 20].

Table 1. Comparisons

Notation	Algorithm Description
Katz-VNE	Heuristic energy-saving virtual network embedding algorithm based on the Katz centrality
EA-VNE	Energy-saving virtual network embedding algorithm based on the shortest path
D-ViNE	Non-energy-saving virtual network embedding algorithm based on the shortest path, adding meta-nodes and meta-paths to the substrate nodes to form local meta-maps for virtual node and link selection
JLB-VNE	Joint load balancing and energy saving algorithm for virtual network embedding

6.2. Evaluation results

Fig. 1 depicts the comparisons of a VN request acceptance rate. As shown in Fig. 1, due to the Katz centrality and hot spot avoidance factor $\psi(p)$, the Katz-VNE achieves a 0.78 VN request acceptance rate in the long run, which is higher than JLB-VNE, EA-VNE and D-ViNE. In node and link allocation, the Katz-VNE can solve the hot spot limitation of the shortest path method and embed a virtual node onto a substrate node that has higher centrality, considering the relationship between all the network nodes and this node. The EA-VNE and D-ViNE based on the shortest path only consider the influence of the current node, which can generate the hot spots, leading to a lower VN request acceptance rate than the Katz-VNE. Due to joint load balancing and energy saving, the JLB-VNE has a higher VN request acceptance rate than the EA-VNE and D-ViNE.

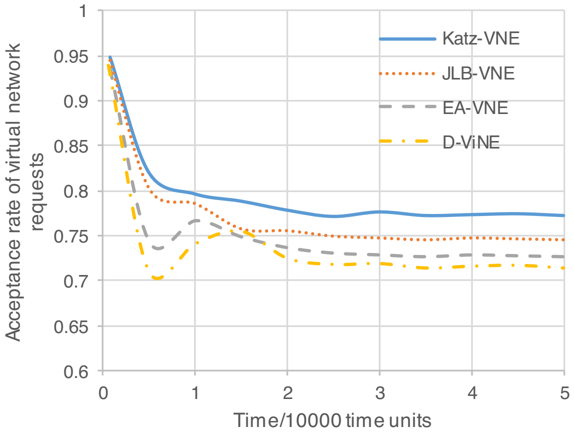

Fig. 1. Acceptance rate of virtual network requests

Fig. 2 depicts the average energy consumption of embedding VNs. Although the Katz-VNE is slightly higher than the EA-VNE from 0 to 10 000 time unit, the Katz-VNE achieves continuously

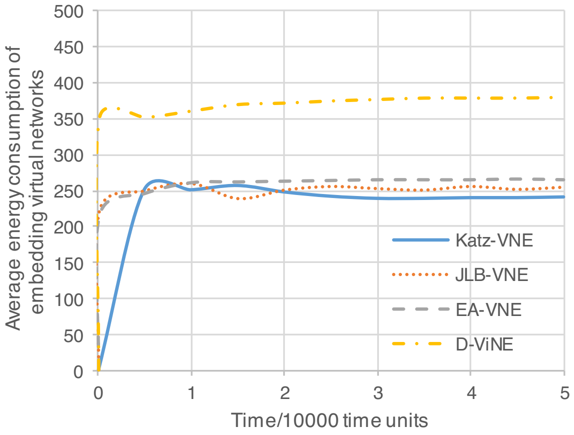

Fig. 2. Average energy consumption of embedding virtual networks

lower average energy consumption after the 10 000 time unit. The Katz-VNE, JLB-VNE, EA-VNE and D-ViNE are stable at 220 W, 235 W, 250 W and 360 W. The reason is that fewer VN requests were reached in the time units from 0 to 10 000. There are many resources for the EA-VNE that are based on the shortest path to embedding. The links used by the EA-VNE are shorter than that of the Katz-VNE, leading to higher average energy consumption of the Katz-VNE. After the 15 000 time unit, the EA-VNE generates more hot spots, and the substrate links allocated by the VN request are gradually increased. The Katz-VNE can solve the hot spot limitation of the shortest path methods and have lower average energy consumption in the long run. Although the JLB-VNE uses a load balancing policy, the average energy consumption is still higher than that of the Katz-VNE.

Fig. 3 shows the long-term revenue/cost ratio of embedding VN requests. Although the Katz-VNE is lower than the EA-VNE from 0 to 6 500 time unit, the Katz-VNE achieves a higher long-term revenue/cost ratio after the 6 500 time unit. The D-ViNE is lower than the Katz-VNE, JLB-VNE, and EA-VNE throughout the experiment. The Katz-VNE, JLB-VNE, EA-VNE and D-ViNE are stable at 0.79, 0.71, 0.69 and 0.54. The reason is that hot spots in the JLB-VNE, EA-VNE and D-ViNE are increasing with the arrival of the VN requests, which is similar to the average energy consumption of embedding VNs shown in Fig. 3.

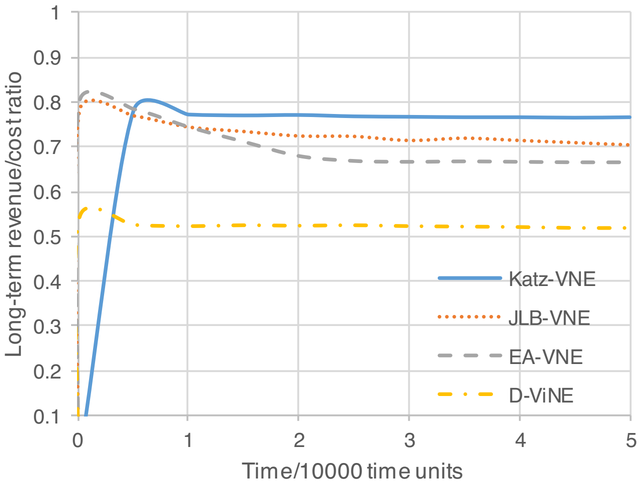

Fig. 3. Long-term revenue/cost ratio

Fig. 4 and Fig. 5 depict the hibernation rate of the SN nodes and links. Fig. 4 and Fig. 5 show that the EA-VNE is slightly higher than the Katz-VNE, JLB-VNE, and D-ViNE in the first 7 000 and 9 000 time units. But the Katz-VNE achieves a substantially and continuously higher hibernation rate than the JLB-VNE, EA-VNE and D-ViNE in the other time units. The reason is that the EA-VNE based on the shortest path can get better experimental results. When it arrives continuously with the VN requests, the number of hot spots increases, leading to the lower experimental results.

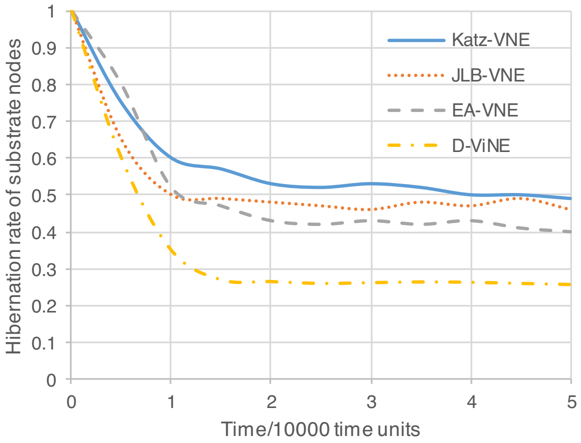

Fig. 4. Hibernation rate of substrate nodes

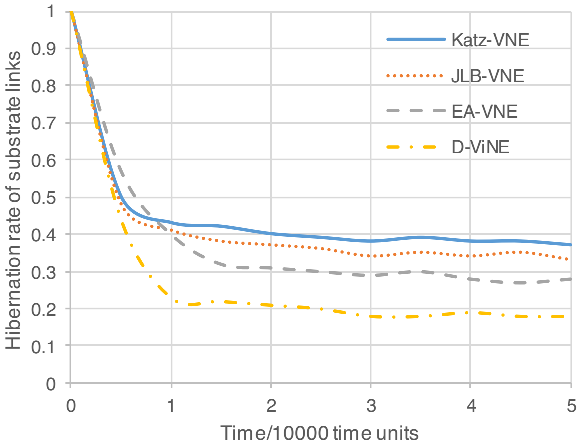

Fig. 5. Hibernation rate of substrate links

7. Conclusions

The current SN equipment designed for peak loads, causes a great waste of power resources. An ES-VNE algorithm is important in reducing the energy consumption of the SN and improves the acceptance rate as much as possible in the long run. Unlike the traditional algorithms based on the shortest path, we have proposed the Katz-VNE by introducing the concept of node centrality in social networks. The Katz-VNE tries to select substrate nodes with high Katz centrality, which works by using hot spot avoidance factor to avoid hot nodes. The simulation results demonstrate that the Katz-VNE is better than the three compared algorithms. We will focus on the efficiency of the allocation strategy to reduce the energy consumption and apply to the electrified fence in the future.

Acknowledgements

This paper was supported by the National Natural Science Foundation of China under Grant (No. 61741311), the natural science foundation of Shandong Province (No. ZR2017PF010), the Yantai science and technology project (No. 2016ZH061).

References

[1] Yun D., Lee J., *Research in green network for future internet*, Journal of Korean Institute of Information Scientists and Engineers, vol. 28, no. 1, pp. 41–51 (2010).

[2] Anderson T., Peterson L., Shenker S., *et al.*, *Overcoming the Internet impasse through virtualization*, Computer, vol. 38, no. 4, pp. 34–41 (2005).

[3] Feamster N., Gao L., Rexford J., *How to lease the Internet in your spare time*, ACM SIGCOMM Computer Communication Review, vol. 37, no. 1, pp. 61–64 (2007).

[4] Chowdhury N.M.M.K., Boutaba R., *A survey of network virtualization*, Computer Networks, vol. 54, no. 5, pp. 862–876 (2010).

[5] Chowdhury N.M.M.K., Rahman M.R., Boutaba B., *ViNEYard: embedding algorithms with coordinated node and link mapping*, IEEE/ACM Transactions on Networking, vol. 20, no. 1, pp. 206–219 (2012).

[6] Cheng X., Su S., Zhang Z.B., Wang H.C., Yang F.C., Luo Y., Wang J., *Virtual network embedding through topology-aware node ranking*, ACM SIGCOMM Computer Communication Review, vol. 41, no. 2, pp. 39–47 (2011).

[7] Rahman M., Boutaba R., *SVNE: survivable virtual network embedding algorithms for network virtualization*, IEEE Transactions on Network and Service Management, vol. 10, no. 2, pp. 105–118 (2013).

[8] Yeow W.L., Westpha C., Kozatl U.C., *Designing and embedding reliable virtual infrastructures*, ACM SIGCOMM Computer Communication Review, vol. 41, no. 2, pp. 57–64 (2011).

[9] Zhu Q., Wang H.Q., Feng G.S. *et al.*, *A hybrid reliable heuristic mapping method based on survivable virtual networks for network virtualization*, Discrete Dynamics in Nature and Society (2015).

[10] Botero J.F., Hesselbach X., Duelli M. *et al.*, *Energy efficient virtual network embedding*, Communications Letters, vol. 16, no. 5, pp. 756–759 (2012).

[11] Botero J.F., Hesselbach X., *Greener networking in a network virtualization environment*, Computer Networks, vol. 57, no. 9, pp. 2021–2039 (2013).

[12] Wang B., Chang X., Liu J. *et al.*, *Reducing power consumption in embedding virtual infrastructures*, Proceedings of 2012 IEEE Global Communications Conference Workshops, IEEE, Anaheim, California, USA, pp. 714–718 (2012).

[13] Sun G., Anan V., Liao D. *et al.*, *Power-efficient provisioning for online virtual network requests in cloud-based data centers*, IEEE Systems Journal, vol. 9, no. 2, pp. 427–441 (2015).

[14] Chen X.H., Li C.Z., Chen L.Y., Zeng Z.B., *Energy Efficient Virtual Network Embedding Based on Actively Hibernating Substrate Nodes and Links*, Journal of Software, vol. 25, no. 7, pp. 1416–1431 (2014).

[15] Liao S.Q., Wu C.M., Wang B. *et al.*, *Research on energy-saving method based on reconfigurable network architecture*, Journal on Communications, vol. 33, no. 9, pp. 77–84 (2012).

[16] Chen X.H., Li C.Z., Jiang Y.L., *Optimization model and algorithm for energy efficient virtual node embedding*, IEEE Communication Letters, vol. 19, no. 8, pp. 1327–1330 (2015).

[17] Zhang Z., Liu A.X., Cheng X. *et al.*, *Energy-aware virtual network embedding*, IEEE/ACM Transactions on Networking, vol. 22, no. 5, pp. 1607–1620 (2014).

[18] Zheng X.W., Hu B., Lu D.J. *et al*, *Energy-efficient virtual network embedding in networks for cloud computing*, International Journal of Web and Grid Services, vol. 13, no. 1, pp. 75–93 (2017).

[19] Fan X., Weber W.D., Barroso L.A., *Power provisioning for a warehouse-sized computer*, Proceedings of 2007 ACM SIGARCH computer architecture news, ACM, New York, USA, vol. 35, no. 2, pp. 13–23 (2007).

[20] Pyoung C.K., Baek S.J., *Joint load balancing and energy saving algorithm for virtual network embedding in infrastructure providers*, Computer Communications, vol. 121, pp. 1–18 (2018).

[21] Tang H., Wu J., *Multi-objective coordination optimisation method for DGs and EVs in distribution networks*, Archives of Electrical Engineering, vol. 68, no. 1, pp. 15–32 (2019).

[22] Singh H.P., Brar Y.S., Kothari D.P., *Reactive power based fair calculation approach for multiobjective load dispatch problem*, Archives of Electrical Engineering, vol. 68, no. 4, pp. 719–735 (2019).