Frontiers of Information Technology & Electronic Engineering
www.zju.edu.cn/jzus; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Virtual network embedding based on real-time topological attributes*

Jian DING†, Tao HUANG, Jiang LIU, Yun-jie LIU

(*State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China*)

†E-mail: buptsky@gmail.com

Received Apr. 23, 2014; Revision accepted Sept. 8, 2014; Crosschecked Jan. 5, 2015

Abstract: As a great challenge of network virtualization, virtual network embedding/mapping is increasingly important. It aims to successfully and efficiently assign the nodes and links of a virtual network (VN) onto a shared substrate network. The problem has been proved to be NP-hard and some heuristic algorithms have been proposed. However, most of the algorithms use only the local information of a node, such as CPU capacity and bandwidth, to determine how to map a VN, without considering the topological attributes which may pose significant impact on the performance of the embedding. In this paper, a new embedding algorithm is proposed based on real-time topological attributes. The concept of betweenness centrality in graph theory is borrowed to sort the nodes of VNs, and the nodes of the substrate network are sorted according to the correlation properties between the former selected and unselected nodes. In this way, node mapping and link mapping can be well coupled. A simulator is built to evaluate the performance of the proposed virtual network embedding (VNE) algorithm. The results show that the new algorithm significantly increases the revenue/cost (R/C) ratio and acceptance ratio as well as reduces the runtime.

Key words: Virtual network embedding (VNE), Real-time topological attributes, Betweenness centrality, Correlation properties, Network virtualization
doi:10.1631/FITEE.1400147 **Document code:** A **CLC number:** TP393

1 Introduction

Network virtualization has been put forward as one of the underlying technologies for future Internet (Anderson *et al.*, 2005; Bavier *et al.*, 2006; Fischer *et al.*, 2013). It aims to enable multiple virtual networks to run on a shared physical substrate and provide a variety of customized services. Thus, multiple service providers (SPs) are able to dynamically compose different heterogeneous VNs to deploy customized end-to-end services using substrate network resources managed by infrastructure providers (INPs). It has already been applied to testbeds such as 4WARD and the cloud computing environment to assess new network protocols and services (Fischer *et al.*, 2013).

The virtual network embedding (VNE) problem is a fundamental part of network virtualization. The main purpose of the VNE problem is to dynamically allocate the virtual network onto the physical substrate hardware efficiently on the basis of satisfying the constraints of nodes and links, while maximizing the benefits obtained from existing hardware.

Due to the constraints of nodes and links in both virtual network (VN) requests and the substrate networks, the VNE problem has been proved to be NP-hard (Andersen, 2002). Thus, present work mainly attempts to design heuristic algorithms on the VNE problem. However, most of them measure only the local resource of nodes (Ricci *et al.*, 2003; Zhu

* Project supported by the National Basic Research Program (973) of China (No. 2012CB315801), the National Natural Science Foundation of China (Nos. 61302089 and 61300184), and the Fundamental Research Funds for the Central Universities, China (No. 2013RC0113)
ORCID: Jian DING, http://orcid.org/0000-0002-9147-8119; Jiang LIU, http://orcid.org/0000-0002-0729-1299

and Ammar, 2006; Yu *et al.*, 2008), such as CPU capacity and bandwidth, without considering topological attributes which may have significant influence on the performance of embedding. Wang *et al.* (2012) measured the resource of the nodes in both VNs and substrate networks by introducing closeness centrality into the VNE problem. However, the attributes of the nodes in VN requests and substrate networks are not identical due to their different roles in the VNE problem. They should not be measured in the same way.

In this paper, in contrast to previous solutions, we consider not only the local network resource but also real-time topological attributes when ranking the importance of nodes of VNs and substrate networks. The concept of 'betweenness centrality' in graph theory is brought into the VNE problem for the analysis of nodes in the virtual request, and the correlation properties between the former selected and unselected nodes are considered in the substrate networks during the mapping process. Based on the nodes ranking results of VNs and substrate networks, we propose a new VNE algorithm called RTA-MAX based on the real-time topological attributes to successfully and efficiently map VNs to a substrate network. RTA-MAX is a two-stage VNE algorithm, which maps virtual nodes to the substrate nodes according to the rankings of node importance in the node mapping stage, and then maps the virtual links using the k-shortest path algorithm (Liu *et al.*, 2011).

A simulator model is developed to evaluate the performance of the proposed algorithm. Compared with the existing algorithms, our new algorithm performs much better in terms of the acceptance ratio, revenue/cost ratio, and runtime.

The major contributions presented in this paper are summarized as follows:

1. As far as we know, this is the first work to employ the network betweenness concept in graph theory into the VNE problem. In the embedding process, we rank the significance of the nodes in virtual network requests by both topology-aware betweenness centrality from network betweenness analysis and the local resource, which makes the bridge nodes with large nodal flow (the flow passing through a node) in VN requests be mapped preferentially.

2. The correlation properties of the former selected and unselected nodes are introduced to rank the substrate nodes during embedding. We take into consideration the correlation properties in the measurement of substrate nodes resource during embedding, which is very favorable for the following link mapping.

2 Related work

The VNE problem is an NP-hard problem even if some constraints are ignored. Some researchers restricted the problem space by considering the constraints of links only and assuming that the node mapping is already known (Ricci *et al.*, 2003; Lu and Turner, 2006; Zhu and Ammar, 2006). Offline approaches were presented in Fan and Ammar (2006), Lu and Turner (2006), and Zhu and Ammar (2006), where all the VN requests are known before embedding. Some previous work did not consider the admission control problem with insufficient resource (Fan and Ammar, 2006; Lu and Turner, 2006; Zhu and Ammar, 2006). Yu *et al.* (2008) introduced a mechanism that supports path splitting and migration in substrate and a multi-commodity flow algorithm to map virtual links to improve long-term average revenue, which considers both online process and admission control. Chowdhury *et al.* (2009) proposed a new algorithm by building an augmented graph with meta-nodes and solved the mapping problem based on a mixed integer programming algorithm. Cheng *et al.* (2011; 2012) applied the Markov random walk to rank network nodes, and extended their work by introducing an integer linear programming formulation to optimize the embedding solution after ranking the nodes in a network. They measured the topological attributes by considering the available resources of adjacent nodes and defining bias factors to express the probability of a node's next movement in the Markov random walk. However, they did not consider the attributes of link bandwidth in resource measurement. An algorithm based on the proximity principle which considers the distance factor during node mapping was proposed in Liu *et al.* (2011). Li *et al.* (2012) considered the topology of the VN request in the node mapping stage, which intends to make the nodes connected with each other in VN topology be mapped close in substrate. They selected the substrate node with fewer hops of substrate path to the former mapped VN nodes for a VN node. However, they did not consider the available bandwidth of paths between the selected substrate nodes and may fail to map the virtual nodes close when the constraint of links is not satisfied in the link mapping stage. Closeness centrality was introduced into the problem in Wang *et al.* (2012), which measured the rank of nodes by considering the closeness of nodes in both virtual and substrate networks.

In this paper, we present a new algorithm based on real-time topological attributes for the VNE problem. In contrast to previous work, we consider the attribute of VNs and the substrate network in different ways since they have different roles in embedding. Both betweeness centrality and local resource of nodes are considered in virtual network analysis to sort the virtual nodes before node mapping. We measure the substrate node with local resource, including CPU capacity and connected links' bandwidth, closeness centrality, and the correlation properties between the former selected and unselected nodes. The model, based on Yu *et al.* (2008), is also adopted to make our work closer to a practical embedding system.

3 Virtual network embedding problem

In this section, the VNE problem is presented first, including general definitions and formulations. Then we introduce the objectives of the VNE problem.

3.1 Substrate network

The substrate network can be modeled as an undirected weighted graph $G_S=(N_S, E_S, C_N^S, C_L^S)$, where N_S is the set of nodes and E_S the set of links in the substrate. C_N^S and C_L^S denote CPU capacity of nodes and bandwidth of links, respectively. Like most previous research, we consider the available CPU capacity as the node attribute and the available bandwidth as the link attribute. An example of substrate network is presented in Fig. 1.

3.2 Virtual network request

Each virtual network request is modeled as an undirected weighted graph $G_V=(N_V, E_V, C_N^V, C_L^V)$ similar to that of the substrate network, where N_V is the set of nodes and E_V the set of links in the VN request. C_N^V and C_L^V denote the constraints of the

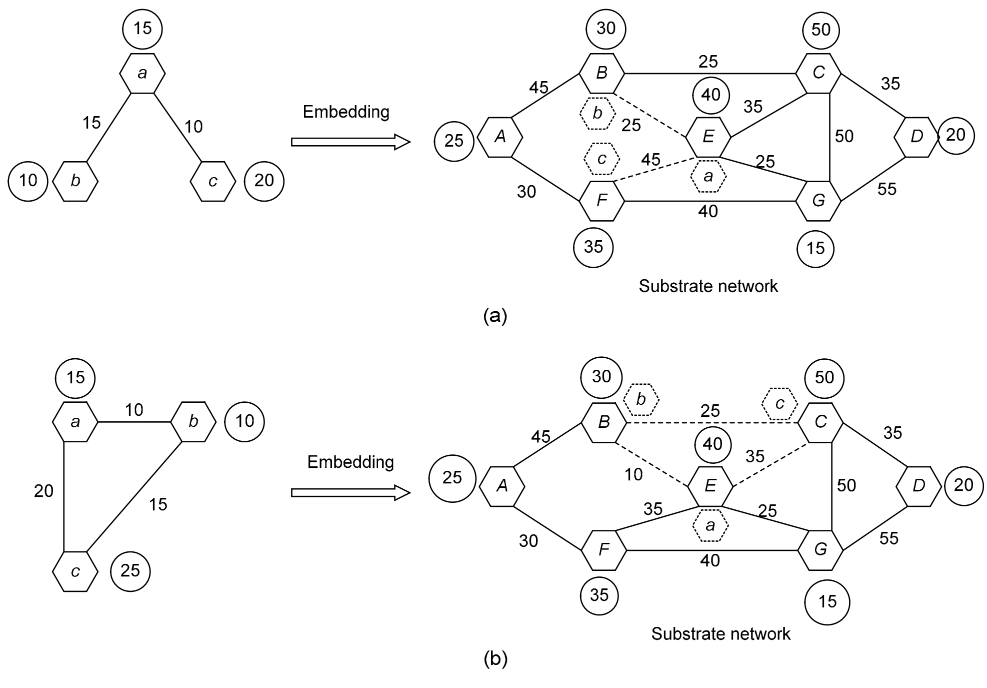

Fig. 1 Mapping virtual networks (VNs) to a shared substrate network: (a) VN request 1; (b) VN request 2
The numbers of available CPU resources of the nodes are given inside circles and the numbers of available bandwidths of the links are beside the edges

nodes and links in the VN requests, respectively. Two VN request examples are presented in Fig. 1.

3.3 Virtual network embedding problem

The embedding problem focuses mainly on how to efficiently allocate a virtual network onto the physical substrate hardware with the constraints of nodes and links satisfied, while gaining maximum benefit from the existing hardware source. As shown in Fig. 1, two VN requests supplied by two different SPs are assigned to a shared physical substrate network, which is managed by an infrastructure provider. SPs will be able to provide multiple services to end users on the same infrastructure in the network virtualization environment with VNE technology.

The VNE problem is defined as an embedding action M from G_{V} to a subset of the substrate G_{S} while the constraints of nodes and links in G_{V} are satisfied, which can be defined as follows:

$$M(G_{\mathrm{V}}):(N_{\mathrm{V}},E_{\mathrm{V}},C_{\mathrm{N}}^{\mathrm{V}},C_{\mathrm{L}}^{\mathrm{V}})\rightarrow(N_{\mathrm{S}},E_{\mathrm{S}},C_{\mathrm{N}}^{\mathrm{S}},C_{\mathrm{L}}^{\mathrm{S}}).$$

As shown above, the VNE problem breaks apart into two steps: (1) mapping virtual nodes to certain nodes in the substrate while the resource constraints of nodes are satisfied; (2) allocating virtual links to the available path in the substrate while the resource requirement of links is satisfied.

3.4 Metrics for evaluating the VNE problem

VNE is an NP-hard optimization problem to efficiently allocate the VNs to the shared substrate network.

Revenue of the VNE problem is the benefit gained by successfully allocating a VN request. Similar to earlier work (Zhu and Ammar, 2006; Yu *et al.*, 2008; Chowdhury *et al.*, 2009), we formulate the revenue of a VN request G_{V} at time t as follows:

$$R(M(G_{\mathrm{V}}),t)=\sum_{e_{\mathrm{V}}\in E_{\mathrm{V}}}\mathrm{BW}(e_{\mathrm{V}})+\sum_{n_{\mathrm{V}}\in N_{\mathrm{V}}}\mathrm{CPU}(n_{\mathrm{V}}),\quad(1)$$

where $\mathrm{BW}(e_{\mathrm{V}})$ is the bandwidth requirement of virtual edge e_{V}, and $\mathrm{CPU}(n_{\mathrm{V}})$ is the CPU requirement of virtual node n_{V}. We define the long-term average revenue as

$$\lim_{T\rightarrow\infty}\frac{1}{T}\sum_{t=0}^{T}R(M(G_{\mathrm{V}}),t).$$

Cost of the VNE problem is the price for embedding the VN request. It is formulated according to the CPU of nodes and bandwidth of links occupied by the VN in the substrate network:

$$\begin{aligned}C(M(G_{\mathrm{V}}),t)=&\sum_{e_{\mathrm{V}}\in E_{\mathrm{V}}}\mathrm{HOP}(e_{\mathrm{V}})\cdot\mathrm{BW}(e_{\mathrm{V}})\\&+\sum_{n_{\mathrm{V}}\in N_{\mathrm{V}}}\mathrm{CPU}(n_{\mathrm{V}}),\end{aligned}\quad(2)$$

where $\mathrm{HOP}(e_{\mathrm{V}})$ is the hop number of a set of substrate links, to which the virtual link e_{V} is assigned. Note that path splitting is not allowed in this model and the CPU cost of intermediate nodes, which is cost to forward data in a multi-hop path for a virtual link, is contained by the BW part in this widely used cost model.

The long-term average revenue is defined as

$$\lim_{T\rightarrow\infty}\frac{1}{T}\sum_{t=0}^{T}C(M(G_{\mathrm{V}}),t).$$

The ratio between revenue and cost is significant in reflecting the performance of network resource utilization for a VNE algorithm. We define the ratio as

$$\mathrm{R/C}=\frac{\sum_{e_{\mathrm{V}}\in E_{\mathrm{V}}}\mathrm{BW}(e_{\mathrm{V}})+\sum_{n_{\mathrm{V}}\in N_{\mathrm{V}}}\mathrm{CPU}(n_{\mathrm{V}})}{\sum_{e_{\mathrm{V}}\in E_{\mathrm{V}}}\mathrm{HOP}(e_{\mathrm{V}})\cdot\mathrm{BW}(e_{\mathrm{V}})+\sum_{n_{\mathrm{V}}\in N_{\mathrm{V}}}\mathrm{CPU}(n_{\mathrm{V}})}.\quad(3)$$

The acceptance ratio, which indicates the number of accepted VNs from all the VNs, reflects the success rate of a VNE algorithm. It is defined as

$$\text{acceptance ratio}=\frac{\text{accepted VN requests count}}{\text{all VN requests count}}.\quad(4)$$

4 Real-time topological attributes analysis

In this section, we first analyze the use of network centrality in the VN problem. Then we explain the correlation properties between the mapped and unmapped nodes in the substrate.

4.1 Network centrality

Network centrality is an important topological attribute in network analysis to indicate the significance of elements in a network, such as nodes, links, and even the whole network. Recently, it has been proposed as a measurement of the importance, popularity, and prominence of a node in social network studies. In graph theory, the measure of node centrality is divided into various kinds according to the generation principle, based on connection, shortest path, flow, random walk, and feedback, respectively. Three algorithms are widely used in network analysis: degree centrality based on connection, closeness and betweenness centralities both based on the shortest path. These different algorithms indicate different attributes of the nodes in a network from different points of view, and result in different node ranks. Degree centrality focuses on measuring the local centrality of a node by considering only the number of nodes directly connected to it, which is too partial in VNE problem analysis.

Inspired by previous work (Wang *et al.*, 2012), the closeness centrality is defined by the sum of the distance from a node to all the other nodes in a network on a global scale, which reflects how close the node is to the other nodes. Thus, ranking nodes by closeness centrality may not find the nodes in the core position of the network but the ones which are very close to all the other nodes. So, we take into account closeness centrality to measure the importance of substrate nodes during the node mapping step, which is very helpful in improving the success rate and source utilization efficiency in the link mapping process. Since the VNE problem is a real-time network analysis with state of nodes and links, rather than simple graph theory, the bandwidth of links and CPU of nodes are also considered. The closeness centrality in VNE can be modified as

$$C(n_i)=\sum_{j=1}^{n}c_j\cdot\frac{\min(\mathrm{bw}(i,j))}{d(i,j)},\tag{5}$$

where $C(n_i)$ is the closeness centrality of substrate node n_i, c_j refers to the CPU capacity of substrate node n_j, $\min(\mathrm{bw}(i, j))$ is the real-time available bandwidth on the shortest path from n_i to n_j, and it denotes the state of links in real time. We describe the minimum bandwidth along the shortest path as the available capacity. $d(i, j)$ is the distance from node n_i to n_j, and here is described by the hop count along the shortest path between them. Fig. 2 shows an example of VNE, where a VN request with three virtual nodes *a*, *b*, and *c* is mapped to a substrate network with five nodes *A*, *B*, *C*, *D*, and *E*. As defined above, the closeness centrality of node *C* in the substrate is determined by the CPU of the other four nodes and the available bandwidth of the shortest paths between them, including the paths *C*-*D*-*A*, *C*-*B*, *C*-*D*, and *C*-*E*.

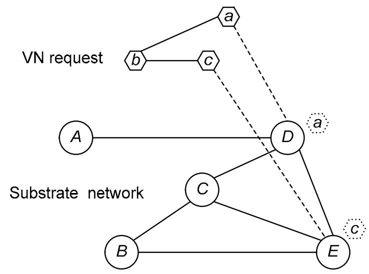

Fig. 2 An example of virtual network embedding (VNE)

Node ranking in the substrate network focuses on the distance which is helpful for link mapping. However, the bridge nodes in VN requests which may carry more nodal flow should be mapped first since early mapped nodes have more available resources and better locations to choose in the substrate network. The betweenness centrality of a node is defined as the number of shortest paths from all nodes to all others that pass through the node, which is always proposed to measure the load and importance of a node in a social network. Inspired by this idea, we select betweenness centrality to measure the nodes in VN requests and define it as

$$B(n_i)=\frac{s_i}{S},\tag{6}$$

where $B(n_i)$ is the betweenness centrality of virtual node n_i, s_i denotes the number of the shortest paths from all nodes to all others in the whole virtual network that pass through node n_i, and S is the number of all the shortest paths. As shown in Fig. 2, the shortest paths in the VN include *a*-*b*, *a*-*b*-*c*, and *b*-*c*, and two of them pass through node *c*. Thus, $B(c)$ is 2/3.

4.2 Correlation property

In previous work, nodes of VNs are mapped to the substrate network one by one separately, and only the attributes of unmapped nodes are considered. The selected nodes in the substrate may be far away from each other with multiple hops and result in inefficient utilization of physical network resources because one virtual link will be mapped to multiple substrate links. To overcome this problem, we take into account the correlation property between the mapped and unmapped nodes in node ranking of the substrate network based on both distance and bandwidth. We define it as follows:

$$\mathrm{Pro}(n_i) = \gamma^{\sum_{j=1}^{k} \min(\mathrm{bw}(i,j))/d(i,j)}, \tag{7}$$

where $\mathrm{Pro}(n_i)$ denotes the correlation property between an unselected node n_i and all the former selected nodes in the substrate, γ is Euler's constant. The correlation property of the unselected substrate node n_i is codetermined by the available bandwidth and distance from itself to all the former selected nodes, which will keep the selected nodes connected closely to each other with proper available bandwidth and is favorable for the following link mapping. As shown in Fig. 2, virtual nodes a and c have already been embedded to nodes D and E, respectively. $\mathrm{Pro}(C)$ is determined by $\min(\mathrm{bw}(C, D))$, which is the available bandwidth of link C-D, and $\min(\mathrm{bw}(C, E))$, which is the available bandwidth of link C-E, while both $d(C, D)$ and $d(C, E)$ are 1.

5 A new VNE algorithm based on real-time topological attributes

On the basis of real-time topological attributes analysis in the network, we devise a new two-stage VNE algorithm called RTA-MAX (Algorithms 1 and 2) to improve the utilization of physical resource and mapping performance. It is shown in the performance evaluation of Section 6 that the results of the acceptance ratio, resource cost, and mapping runtime become much better. The detailed description of the algorithm is as follows: sort the VN requests according to their revenues in descending order, and select the VN request with the maximum revenue to embed. Then we begin the node mapping stage and sort the virtual nodes in this VN by considering both the required local resources and betweenness centrality (Section 4.1) of the nodes, which is formulated as

$$\mathrm{Value}_{\mathrm{v}}(n_{\mathrm{v}}) = \mathrm{CPU}(n_{\mathrm{v}}) \cdot \sum_{l_{\mathrm{v}} \in L(n_{\mathrm{v}})} \mathrm{BW}(l_{\mathrm{v}}) \cdot B(n_{\mathrm{v}}), \tag{8}$$

where $\mathrm{Value}_{\mathrm{v}}(n_{\mathrm{v}})$ is the factor to sort the virtual node n_{v}, $\mathrm{CPU}(n_{\mathrm{v}})$ denotes the required CPU of n_{v}, $L(n_{\mathrm{v}})$ denotes the set of connected virtual links of n_{v}, and $B(n_{\mathrm{v}})$ is the betweenness centrality of n_{v} introduced in Section 4.1.

Then we define the variable $\mathrm{Value}_{\mathrm{s}}$ to sort the substrate nodes by considering closeness centrality (Section 4.1) and correlation properties (Section 4.2) of the nodes:

$$\mathrm{Value}_{\mathrm{s}}(n_{\mathrm{s}}) = \mathrm{CPU}(n_{\mathrm{s}}) \cdot \sum_{l_{\mathrm{s}} \in L(n_{\mathrm{s}})} \mathrm{BW}(l_{\mathrm{s}}) \cdot C(n_{\mathrm{s}}) \cdot \mathrm{Pro}(n_{\mathrm{s}}), \tag{9}$$

where $\mathrm{Value}_{\mathrm{s}}(n_{\mathrm{s}})$ is the factor to rank the substrate node n_{s}, $\mathrm{CPU}(n_{\mathrm{s}})$ denotes the required CPU of n_{s}, $L(n_{\mathrm{s}})$ denotes the set of connected physical links of n_{s}, $C(n_{\mathrm{s}})$ is the closeness centrality of n_{s}, and $\mathrm{Pro}(n_{\mathrm{s}})$ is the correlation properties.

Algorithm 1 Node mapping stage of RTA-MAX

```
Sort the VN requests by their revenues in decreasing order
for each VN request in the order do
  Sort the nodes in VN request according to their Value_v
    values in decreasing order
  for each unmapped virtual node in the order do
    Sort the nodes in substrate network according to their
      Value_v values in decreasing order
    Map the virtual node to the substrate node which is un-
      mapped and with the largest Value_s, with the required
      source also being satisfied
    if the virtual node cannot be mapped then
      return Node unsuccessfully mapped
    end if
  end for
end for
return Node successfully mapped
```

Algorithm 2 Link mapping stage of RTA-MAX

```
for each unmapped virtual link do
  Search the k-shortest paths between the selected nodes in
    substrate
  if find a path with proper bandwidth which satisfies the
    required link resource
    Assign the virtual link to this path
    return Link successfully mapped
  else
    return Link unsuccessfully mapped
  end if
end for
```

We select the virtual node with maximum Value_v and allocate it to the substrate node with both enough required resource and maximum Value_s, which results in nodes with proper resource being selected and makes it easy to map virtual links with each other.

After selecting the substrate nodes, we begin the link mapping stage using the k-shortest path algorithm similar to previous work, which searches the fit shortest paths by filtrating the k-shortest paths until finding a path in the substrate network with proper resource required by the virtual link.

Note that although we add the amount of calculation in the node mapping stage, the whole runtime of RTA-MAX is decreasing in practice, since the algorithm takes into account the real-time topological attributes and results in a better node mapping which is beneficial for the following link mapping stage. We will certify it in the following section by comparing the runtime of different algorithms in the same PC.

6 Performance evaluation

In this section, we first introduce our simulation environment. Then we present our evaluation results and analyze the performance of RTA-MAX in comparison with previous algorithms.

6.1 Simulation environment

We have developed a VNE simulator model to evaluate our new algorithm based on previous work (Yu *et al.*, 2008). The substrate network topology is generated by the GT-ITM tool (Zegura *et al.*, 1996), which is widely used in network topology generation and simulators. The substrate network is generated with 100 nodes and about 500 links, which is approximately the scale of medium-size INPs. The CPU of nodes and bandwidth of links in substrate follow a uniform distribution from 50 to 100.

We adopted the method proposed by Zhu and Ammar (2006) to generate the virtual network requests. The number of nodes in each VN request follows a uniform distribution between 2 and 10. Each pair of virtual nodes is randomly connected to each other with a probability of 50%. The runtime of each VN request is exponentially distributed with a mean of 1000 time units, and the VN requests arrive following a Poisson distribution with a mean of five requests in 100 time units. The CPU of nodes and bandwidth of links in a VN request follow a uniform distribution with the scale increasing from 10% to 90% of the substrate network resource. We ran the simulation under each different condition of the VN requests for 50 000 time units and took the average values to obtain an evaluation result of a stable state. According to the runtime of each VN request, the simulation environment will reach a stable state after running 10 000 time units, which ensures that 50 000 time units is long enough to obtain a stable result.

The metrics we used to evaluate the performance of the VNE algorithm are as follows: R/C ratio, VN acceptance ratio (Eqs. (3) and (4)), and the runtime of the algorithms on the same PC. Four algorithms were evaluated in our simulations (Table 1). Eighteen instances were run for these algorithms under different conditions of the VN requests to evaluate the performance of algorithms and the influence of the increasing size of VN requests.

Table 1 Algorithm comparison

Notation	Description of the algorithm
Greedy	The classical greedy algorithm, with mapping nodes with greedy local resource and links with the k-shortest algorithm (Yu *et al.*, 2008)
CL	Mapping nodes by ranking the nodes based on classical closeness (Wang *et al.*, 2012)
IC	Mapping nodes by ranking the nodes based on improved closeness (Wang *et al.*, 2012)
RTA-MAX	Mapping nodes by measuring the nodes based on real-time topological attributes considering VNs and substrate in different ways

6.2 Results and analysis

6.2.1 General VNs

For each VN request, the required CPU of virtual nodes and BW of virtual links follow a uniform distribution from 0 to 50. Fig. 3 shows the ratio between revenue and cost of the four algorithms. The R/C ratio of RTA-MAX (0.792 on average) is much better than those of the other algorithms, at 27% higher than CL (0.625 on average), 18% higher than IC (0.671 on average), and 23% higher than Greedy (0.644 on average).

Fig. 4 shows the VN requests acceptance ratios of the four algorithms. The acceptance ratio of RTA-MAX (0.965 on average) is better than those of the other algorithms, at 3% higher than CL (0.941 on average), 3% higher than IC (0.935 on average), and 16% higher than Greedy (0.829 on average).

6.2.2 Evaluating the impact by increasing the requests' BW

To evaluate the impact of increasing the size of VN requests' bandwidth, we set the bandwidth of virtual links to increase from 10% to 90% of the substrate resource while keeping the CPU of virtual nodes at 50%.

Fig. 5 shows the ratio between revenue and cost of the four algorithms in the stable state. The R/C ratio of RTA-MAX (0.748 on average) is much better than those of the other algorithms, at 19% higher than CL (0.631 on average), 21% higher than IC (0.620 on average), and 22% higher than Greedy (0.611 on average). For each algorithm, the R/C ratio decreases with the increase of VN requests' bandwidth, since VNs with a larger bandwidth requirement are more likely to be assigned to a longer path in the link mapping stage and use more resource in the substrate.

Fig. 6 shows the VN requests acceptance ratios of the four algorithms in the stable state. The acceptance ratio of RTA-MAX (0.904 on average) is better than those of the other algorithms, at 5% higher than CL (0.859 on average), 3% higher than IC (0.876 on average), and 16% higher than Greedy (0.781 on average). Note that the acceptance ratio increases a little (no more than 5%) as the bandwidth increases up to 60%, because the VN requests are mapped in descending order of revenue to increase the utilization of physical resource. As the bandwidth increases, the embedding of the larger VN request may fail in the link mapping stage and resource would be saved for the embedding of the following smaller VN requests. In this way, there exists a chance that more requests with smaller revenue, instead of one larger request, are mapped successfully on the substrate, which results in the acceptance ratio increasing a little. However, the increase of bandwidth has a larger effect on the acceptance ratio, which results in a sharp decrease of the acceptance ratio after the bandwidth increases up to around 60%.

Fig. 7 shows that runtime of the four algorithms in the same PC. The runtime of RTA-MAX (0.778 s on average) is much less than that of the other algorithms, at 45% shorter than CL (1.417 s on average), 34% shorter than IC (1.175 s on average), and 46% shorter than Greedy (1.428 s on average). Similar to the acceptance ratio, the runtime increases sharply

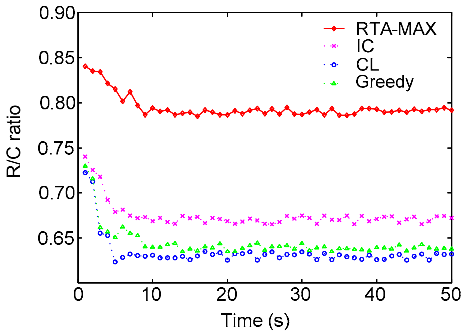

Fig. 3 R/C ratio comparison

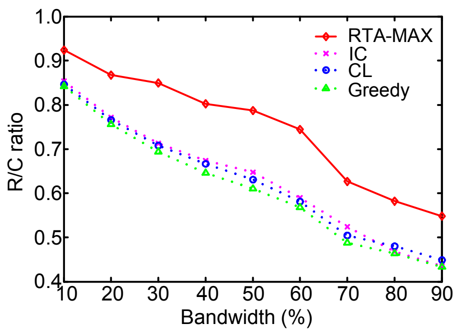

Fig. 5 R/C ratio with increasing bandwidth

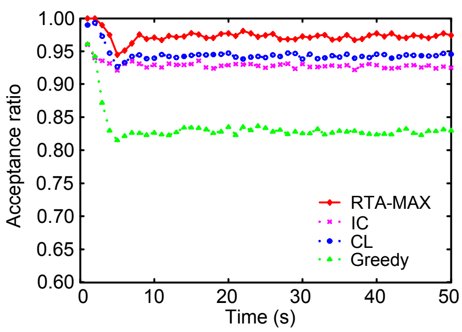

Fig. 4 Acceptance ratio comparison

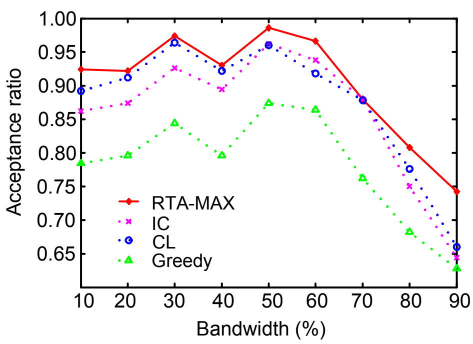

Fig. 6 Acceptance ratio with increasing bandwidth

when the bandwidth increases to 60%, since the embedding algorithm will search the suboptimal path in the *k*-shortest algorithm, when the VNs fail in the link mapping stage.

The evaluation results show that RTA-MAX achieves a much better performance in different BW conditions. This makes a more efficient utilization of substrate physical resource as indicated by the R/C and acceptance ratios. Thus, more VN requests are mapped successfully with higher acceptance ratios which will assist INPs in providing more service to SPs. Furthermore, the runtime is shortened, since the new algorithm makes the mapping stage easier by decreasing backtracking after the node mapping stage considering real-time topological attributes.

6.2.3 Evaluating the impact by increasing the requests' CPU

We increase the CPU of virtual nodes from 10% to 90% of the substrate resource while keeping the bandwidth of virtual links at 50% to evaluate the impact of increasing the size of VN requests' CPU.

Fig. 8 shows the ratio between revenue and cost of the four algorithms. The R/C ratio of RTA-MAX (0.776 on average) is much better than those of the other algorithms, at 33% higher than CL (0.582 on average), 24% higher than IC (0.627 on average), and 27% higher than Greedy (0.612 on average). Note that the R/C ratio increases with the increase of CPU. This is because the value of CPU is the same in the computation of the revenue and cost (Eqs. (1) and (2)). Thus, in the same condition of required bandwidth, the ratio of revenue to cost will increase with the increase of CPU.

Fig. 9 shows the VN requests acceptance ratios of the four algorithms in the stable state. The acceptance ratio of RTA-MAX (0.847 on average) is better than those of the other algorithms, at 3% higher than CL (0.821 on average), 5% higher than IC (0.810 on average), and 11% higher than Greedy (0.763 on average). The acceptance ratio decreases sharply for each algorithm when the required CPU of VNs increases to 40%, which shows that the change of CPU has more impact on the acceptance ratio than BW by comparison with the results in Section 6.2.1.

Fig. 10 shows the runtime of the four algorithms. The runtime of RTA-MAX (0.114 s on average) is much less than that of the other algorithms, which is

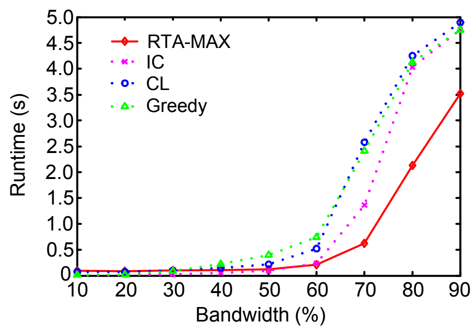

Fig. 7 Runtime with increasing bandwidth

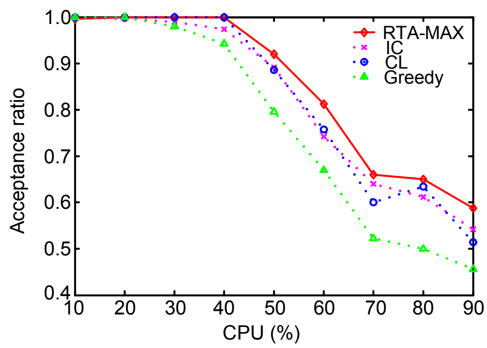

Fig. 9 Acceptance ratio with increasing CPU

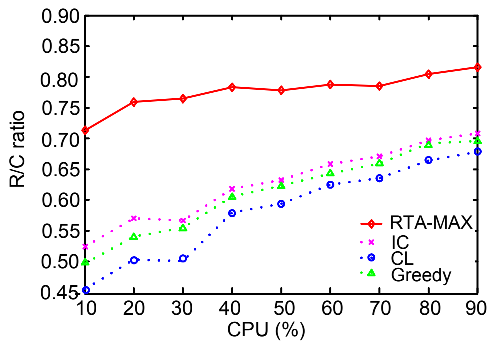

Fig. 8 R/C ratio with increasing CPU

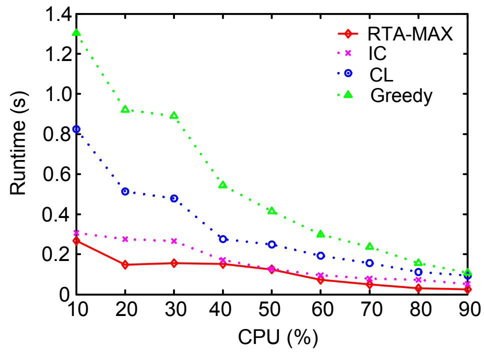

Fig. 10 Runtime with increasing CPU

65% shorter than CL (0.322 s on average), 29% shorter than IC (0.160 s on average), and 79% shorter than Greedy (0.541 s on average). The runtime decreases with the increase of required CPU, since the nodes with larger CPU are more likely to fail in node mapping and the time of link mapping is saved. Note that the runtime with CPU increasing is much shorter than the results with bandwidth increasing as in Section 6.2.1, which indicates that, compared with required CPU, the increase of required bandwidth has more influence on the runtime of the algorithms.

These results show that, in different CPU conditions, our new algorithm RTA-MAX also achieves a much better performance.

7 Conclusions

The virtual network embedding problem is a fundamental part of network virtualization. In this paper, we propose a new algorithm based on real-time topological attributes in the network, which introduces betweenness centrality to analyze the virtual network and the correlation properties for the substrate. Theoretical network analysis and evaluation show that the new algorithm RTA-MAX greatly improves the embedding performance under almost all conditions of both CPU capacity and bandwidth. Moreover, the runtime is reduced.

There are still several issues to be studied. To extend our work, we plan to consider other network attributes, e.g., the node or link fault for a survival environment, the link delay requirement for a special application, and the energy consumption for the green network problem.

References

Andersen, D.G., 2002. Theoretical Approaches to Node Assignment. Available from http://www.cs.cmu.edu/~dga/papers/andersen-assign.ps [Accessed on Sept. 20, 2010].

Anderson, T., Peterson, L., Shenker, S., *et al.*, 2005. Overcoming the Internet impasse through virtualization. *IEEE Comput. Mag.*, **38**(4):34-41.

Bavier, A., Feamster, N., Huang, M., *et al.*, 2006. In VINI veritas: realistic and controlled network experimentation. *ACM SIGCOMM Comput. Commun. Rev.*, **36**(4):3-14. [doi:10.1145/1151659.1159916]

Cheng, X., Su, S., Zhang, Z., *et al.*, 2011. Virtual network embedding through topology-aware node ranking. *ACM SIGCOMM Comput. Commun. Rev.*, **41**(2):38-47. [doi:10.1145/1971162.1971168]

Cheng, X., Su, S., Zhang, Z., *et al.*, 2012. Virtual network embedding through topology awareness and optimization. *Comput. Netw.*, **56**(6):1797-1813. [doi:10.1016/j.comnet.2012.01.022]

Chowdhury, N.M.M.K., Rahman, M.R., Boutaba, R., 2009. Virtual network embedding with coordinated node and link mapping. Proc. 28th IEEE Int. Conf. on Computer Communications, p.783-791. [doi:10.1109/INFCOM.2009.5061987]

Fan, J., Ammar, M.H., 2006. Dynamic topology configuration in service overlay networks: a study of reconfiguration policies. Proc. 25th IEEE Int. Conf. on Computer Communications, p.1-12. [doi:10.1109/INFOCOM.2006.139]

Fischer, A., Botero, J.F., Till Beck, M., *et al.*, 2013. Virtual network embedding: a survey. *IEEE Commun. Surv. Tutor.*, **15**(4):1888-1906. [doi:10.1109/SURV.2013.013013.11055]

Li, X.L., Wang, H.M., Guo, C.G., *et al.*, 2012. Topology awareness algorithm for virtual network mapping. *J. Zhejiang Univ.-Sci. C (Comput. & Electron.)*, **13**(3):178-186. [doi:10.1631/jzus.C1100282]

Liu, J., Huang, T., Chen, J.Y., *et al.*, 2011. A new algorithm based on the proximity principle for the virtual network embedding problem. *J. Zhejiang Univ.-Sci. C (Comput. & Electron.)*, **12**(11):910-918. [doi:10.1631/jzus.C1100003]

Lu, J., Turner, J., 2006. Efficient Mapping of Virtual Networks onto a Shared Substrate. Technical Report No. WUCSE-2006-35, Washington University, USA.

Ricci, R., Alfeld, C., Lepreau, J., 2003. A solver for the network testbed mapping problem. *ACM SIGCOMM Comput. Commun. Rev.*, **33**(2):65-81. [doi:10.1145/956981.956988]

Wang, Z., Han, Y., Lin, T., *et al.*, 2012. Virtual network embedding by exploiting topological information. Proc. IEEE Global Communications Conf., p.2603-2608. [doi:10.1109/GLOCOM.2012.6503509]

Yu, M., Yi, Y., Rexford, J., *et al.*, 2008. Rethinking virtual network embedding: substrate support for path splitting and migration. *ACM SIGCOMM Comput. Commun. Rev.*, **38**(2):17-29. [doi:10.1145/1355734.1355737]

Zegura, E.W., Calvert, K.L., Bhattacharjee, S., 1996. How to model an internetwork. Proc. IEEE 15th Annual Conf. on Computer Communications Jointly with the IEEE Computer and Communications Societies, p.594-602. [doi:10.1109/INFCOM.1996.493353]

Zhu, Y., Ammar, M., 2006. Algorithms for assigning substrate network resources to virtual network components. Proc. 25th IEEE Int. Conf. on Computer Communications, p.1-12. [doi:10.1109/INFOCOM.2006.322]