Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Topology awareness algorithm for virtual network mapping*

Xiao-ling LI[†1,2], Huai-min WANG[1,2], Chang-guo GUO[3], Bo DING[1,2],
Xiao-yong LI[1,2], Wen-qi BI[4], Shuang TAN[2]

([1]*National Key Laboratory of Parallel and Distributed Processing, National University of Defense Technology, Changsha 410073, China*)
([2]*School of Computer, National University of Defense Technology, Changsha 410073, China*)
([3]*China Electronic Systems Engineering Corporation, Beijing 100039, China*)
([4]*The Northern Institute of Electronic Equipment of China, Beijing 100083, China*)
[†]E-mail: nudtlxl@163.com

Received Sept. 26, 2011; Revision accepted Nov. 29, 2011; Crosschecked Feb. 8, 2012

Abstract: Network virtualization is recognized as an effective way to overcome the ossification of the Internet. However, the virtual network mapping problem (VNMP) is a critical challenge, focusing on how to map the virtual networks to the substrate network with efficient utilization of infrastructure resources. The problem can be divided into two phases: node mapping phase and link mapping phase. In the node mapping phase, the existing algorithms usually map those virtual nodes with a complete greedy strategy, without considering the topology among these virtual nodes, resulting in too long substrate paths (with multiple hops). Addressing this problem, we propose a topology awareness mapping algorithm, which considers the topology among these virtual nodes. In the link mapping phase, the new algorithm adopts the *k*-shortest path algorithm. Simulation results show that the new algorithm greatly increases the long-term average revenue, the acceptance ratio, and the long-term revenue-to-cost ratio (*R*/*C*).

Key words: Network virtualization, Ossification, Virtual network (VN) mapping, Substrate network (SN), Topology awareness, Acceptance ratio
doi:10.1631/jzus.C1100282 **Document code:** A **CLC number:** TP393

1 Introduction

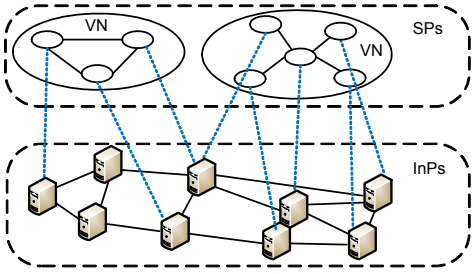
With the rapid development of Internet technology, the existing network architecture is difficult to maintain, given the development of the new diverse applications, and there is a rigid ossification problem (Anderson *et al*., 2005). Due to its multi-provider nature, adopting a modification of the existing one requires consensus among competing stakeholders, which is almost impossible to achieve. Network virtualization allows multiple virtual networks (VNs) to simultaneously run on a shared infrastructure to meet diverse network applications. This has been recognized as an effective way to overcome the ossification of the Internet. For example, it can enable multiple researchers to evaluate new network protocols simultaneously on a shared experiment facility (Anderson *et al*., 2005; Bavier *et al*., 2006; Feamster *et al*., 2007).

The virtual network mapping problem (VNMP) is a key issue in network virtualization, which maps multiple VNs to the substrate network (SN) with virtual nodes and links' constraints being satisfied (Fig. 1) (Fan and Ammar, 2006; Zhu and Ammar, 2006; Yu *et al*., 2008). VNs and the SN are provided by service providers (SPs) and infrastructure providers (InPs), respectively. An efficient mapping scheme would help to improve the substrate resource utilization and avoid congestion in the SN.

In multi-tenant virtualization environments, InPs (e.g., cloud providers) and SPs (e.g., cloud users/tenants) play two decoupled roles: InPs manage the

* Project supported by the National Basic Research Program (973) of China (No. 2011CB302601), the National Natural Science Foundation of China (No. 90818028), and the National High-Tech R&D Program (863) of China (No. 2007AA010301)

infrastructure resources, while SPs create VNs and offer end-to-end services (Turner and Taylor, 2005; Feamster *et al*., 2007; Chowdhury *et al*., 2009a; Bansal *et al*., 2011; Cheng *et al*., 2011). Guo *et al*. (2010) proposed a data center network virtualization architecture called SecondNet. In SecondNet, the unit of resource allocation for multiple tenants in the cloud is referred to as a virtual data center (VDC), which consists of virtual machines (VMs) and virtual links. The VDC resource allocation problem is an application of VNMP, and the main difference is the problem scale. Thus, studying how to improve the efficiency of VNMP has a practical significance; for example, it can be an effective policy to instruct the resource allocation for cloud providers, which is critical to both users' computation needs and the cloud providers' monetary gain in cloud computing.



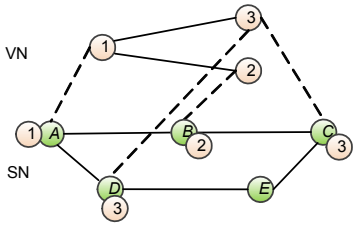
VN: virtual network; SPs: service providers; InPs: infrastructure providers

Fig. 1 Mapping virtual networks to a shared substrate network

In general, the VNMP can be classified into either an offline or an online manner. In the former case, the VN requests are known in advance, and would not change; in the latter case, the VN requests are not known in advance, and are dynamically changeable. For example, a new experiment may run at any time, and then depart for some duration.

Devising heuristic methods has become the main line of research in VNMP. In those methods, the mapping process is always divided into two phases, node mapping phase and link mapping phase, which are responsible for mapping the virtual nodes to substrate nodes and the virtual links to substrate paths, respectively. Several studies have focused on the offline problem. Zhu and Ammar (2006) aimed at achieving load balancing in the SN and obviating the need for admission control. Lu and Turner (2006) considered only a single VN with a backbone-star topology, where their goal is to minimize the cost. Some other studies focus on the online problem. Chowdhury *et al*. (2009b) formulated the VN mapping problem as a mixed integer program (MIP) through SN augmentation, and then relaxed the integer constraints to obtain a linear program. Yu *et al*. (2008) considered node constraints, link constraints, admission control, and the online mapping problem together.

However, the topologies of VNs are also ignored in the mapping phase, which may result in low infrastructure resource utilization. Considering the scenario in Fig. 2, we want to map a VN with three virtual nodes 1, 2, and 3, to an SN with five substrate nodes *A*, *B*, *C*, *D*, and *E*. We first assume that the virtual nodes 1 and 2 have already been mapped to the substrate nodes *A* and *B*, respectively. Moreover, we assume that virtual node 3 has two mapping choices, substrate nodes *C* and *D*. If we do not consider the topology of the VN, we know that the substrate nodes *C* and *D* are all appropriate mapping choices. However, if one considers the topology of the VN, one can know that there is a virtual link between virtual nodes 1 and 3, and there is no virtual link between nodes 2 and 3. Thus, we may easily prefer node *D* to node *C*, because *D* is closer to *A* with respect to *C*, which would be a better choice when one maps the virtual link (1, 3), because the shorter is the substrate path, the less is the use of substrate resource. Thus, when one considers the topologies of the VNs, such as the relations among these nodes, one can obtain better mapping schemes.



VN: virtual network. 1, 2, 3 are three virtual nodes of a VN
SN: substrate network. *A*–*E* are five substrate nodes of an SN

Fig. 2 A mapping scenario that considers the topology of VNs

In this paper, we propose a new topology awareness algorithm for an online VNMP. This algorithm considers the topologies of the VNs in the node mapping phase, which can efficiently reduce the average number of hops of substrate paths and result in high resource utilization.

2 Virtual network mapping problem

In this section, we first describe the general model of VNMP, and then present the problem-solving objectives.

2.1 Formal definition

Definition 1 (Substrate network, SN) We denote SN by an undirected graph $G_S=(V_S, E_S, A_S^V, A_S^E)$, where the subscript S refers to SN, V_S and E_S refer to the sets of substrate nodes and links, respectively, and A_S^V and A_S^E refer to the attributes of substrate nodes and links, respectively.

Similar to Yu *et al*. (2008), Chowdhury *et al*. (2009b), Lischka and Karl (2009), and Wang *et al*. (2009), in this study, we consider only the substrate node's CPU and the substrate link's bandwidth (BW). Thus, A_S^V and A_S^E also refer to CPU consumption and BW, respectively.

$P_S=\{p_1, p_2, \ldots, p_m\}$ denotes the set of substrate paths in SN. For substrate path $p_i \in P_S$, there should have substrate nodes $v_S^{i1}, v_S^{i2}, ..., v_S^{in} \in V_S$ satisfying $p_i=e_S(i1, i2)\rightarrow e_S(i2, i3)\rightarrow\ldots\rightarrow e_S(i(n-1), in)$, where p_i is a substrate path between nodes v_S^{i1} and v_S^{in}, and consists of links $e_S(ij, i(j+1))$, for $0\leqslant j\leqslant n-1$.

Definition 2 (Virtual network, VN) We denote the VN by an undirected graph $G_V=(V_V, E_V, C_V^V, C_V^E)$, where the subscript V refers to VN, V_V and E_V refer to the sets of virtual nodes and links, respectively, and C_V^V and C_V^E refer to the constraints of virtual nodes and links, respectively.

Definition 3 (Virtual network mapping problem, VNMP) VNMP is a mapping process from G_V to G_S, denoted as $M: G_V \mapsto (V_S^*, P_S^*, R_V, R_E)$, where $V_S^* \subset V_S$ and $P_S^* \subset P_S$. In addition, R_V and R_E are the substrate resources allocated to the virtual nodes and virtual links, respectively.

In general, VNMP is composed of the node mapping phase and the link mapping phase. The node mapping phase maps the virtual nodes to the substrate nodes that satisfy the required CPU constraint, and the link mapping phase maps the virtual links to the substrate paths that satisfy the required BW constraint (Lu and Turner, 2006; Zhu and Ammar, 2006).

We denote the node mapping phase by $M^V: (V_V, C_V^V) \mapsto (V_S^*, R_V)$, where $V_S^* \subset V_S$ and R_V are the resources that have been allocated to the virtual nodes. The hard constraint is that different virtual nodes of the same VN cannot be mapped to the same substrate node.

We denote the link mapping phase by M^E: $(E_V, C_V^E) \mapsto (P_S^*, R_E)$, where $P_S^* \subseteq P_S$ and R_E are the resources that have been allocated to the virtual links.

Fig. 3 shows an example of VNMP. Fig. 3c gives a mapping solution for the two VNs in Figs. 3a and 3b. We can easily determine that there are many different mapping solutions for the same problem (for example, mapping the virtual node *b* to the substrate node *A*), and we would obtain some other different mapping solutions.

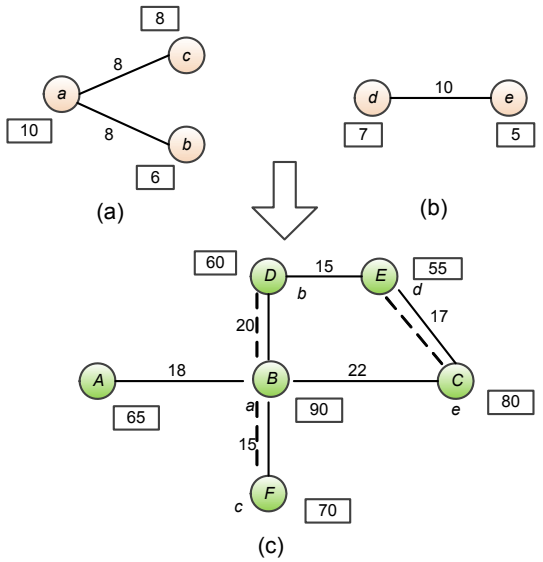


Fig. 3 An example of the virtual network mapping problem (VNMP)

(a) and (b) are two virtual networks, and (c) is a substrate network

2.2 Problem-solving objectives

Similar to Zhu and Ammar (2006), Yu *et al*. (2008), and Cheng *et al*. (2011), the revenue of accepting a VN G_V at time *t* can be defined as

$$R(G_V, t)=\sum_{v_V\in V_V}\mathrm{CPU}(v_V)+\sum_{e_V\in E_V}\mathrm{BW}(e_V), \qquad (1)$$

where CPU(v_V) refers to the required CPU of the virtual node v_V, and BW(e_V) refers to the required

bandwidth of the virtual link e_V. Considering that the relative importance of CPU and BW may be different in different applications, we define a relative factor α to modify Eq. (1), and the revenue of accepting a VN is finally formulated by

$$R(G_V,t)=\sum_{v_V\in V_V}\mathrm{CPU}(v_V)+\alpha\sum_{e_V\in E_V}\mathrm{BW}(e_V). \quad (2)$$

Similar to Yu *et al*. (2008), the cost of accepting a VN G_V at time t can be defined as Eq. (3), which refers to the total substrate resources allocated to the VN:

$$C(G_V,t)=\sum_{v_V\in V_V}\mathrm{CPU}(v_V)+\alpha\sum_{e_V\in E_V}\sum_{e_S\in E_S}\mathrm{BW}(f_{e_S}^{e_V},e_V), \quad (3)$$

where $f_{e_S}^{e_V}\in\{0, 1\}$. When the substrate link e_S is assigned to the virtual link e_V, $f_{e_S}^{e_V}=1$; otherwise, $f_{e_S}^{e_V}=0$. The BW allocated to a virtual link is also equal to the product of the virtual link's required BW and the number of hops of the substrate paths. We also balance the relative importance between CPU and BW with the relative factor α.

Similar to Yu *et al*. (2008) and Cheng *et al*. (2011), the long-term average revenue is defined as

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T}R(G_V,t). \quad (4)$$

Similar to Cheng *et al*. (2011), R/C is defined as the long-term revenue-to-cost ratio, which is formulated by

$$R/C=\lim_{T\to\infty}\frac{\sum_{t=0}^{T}R(G_V,t)}{\sum_{t=0}^{T}C(G_V,t)}. \quad (5)$$

R/C is an important factor for judging the performance of the mapping algorithm, since it indicates the efficiency of infrastructure resources utilization.

In this work, we also consider the access control mechanism. When the substrate resources are not sufficient to be allocated to the new arrival VN at one time, we refuse to accept the VN. Therefore, we define the VN acceptable ratio, similar to Cheng *et al*. (2011), as follows:

$$\lim_{T\to\infty}\frac{\sum_{t=0}^{T}\mathrm{VN_S}}{\sum_{t=0}^{T}\mathrm{VN}}, \quad (6)$$

where $\mathrm{VN_S}$ is the number of VN requests successfully accepted by the SN, and VN is the number of the arrival VN requests.

3 Topology awareness virtual network mapping algorithm

Yu *et al*. (2008) have proposed a simple mapping algorithm (baseline algorithm), which maps virtual nodes using the greedy algorithm and virtual links using the *k*-shortest path algorithm. The baseline algorithm maps the virtual nodes without considering the topology of the VN. As a consequence, virtual nodes connected by the virtual links may be mapped too far away (with multiple hops). Furthermore, according to Eq. (3), we know that the infrastructure resource utilization is low in the baseline algorithm.

Considering the limitation of the baseline algorithm, we propose a new topology awareness mapping algorithm that considers the topologies of the VNs in the node mapping phase.

3.1 Node mapping phase

The baseline algorithm maps the virtual node to the substrate node with the current maximum available resource $H(v_S)$ according to the virtual nodes' required CPU in descending order. However, it does not consider the topology of VN in the node mapping phase. The available resource for substrate node v_S is defined by (Yu *et al*., 2008)

$$H(v_S)=\mathrm{CPU}(v_S)\sum_{l^S\in L(v_S)}\mathrm{BW}(l^S), \quad (7)$$

where $\mathrm{CPU}(v_S)$ refers to the remaining CPU resource of substrate node v_S, $L(v_S)$ is the set of all substrate links that connect v_S, and $\mathrm{BW}(l^S)$ is the unoccupied BW resource for substrate link l^S.

In our algorithm, we first sort the virtual nodes according to their node degrees in descending order. Supposing $m=|V_V|$, without loss of generality, the sorted sequence of virtual nodes can be described as

$$v_{\mathrm{V}}^1, v_{\mathrm{V}}^2, ..., v_{\mathrm{V}}^m,$$

where $DG_1 \geq DG_2 \geq \ldots \geq DG_m$, and DG_i refers to the node degree of virtual node v_{V}^i.

In our algorithm, we map the virtual node to the substrate node with the largest node selection integrated factor (NSIF). NSIF is defined as

$$\mathrm{NSIF}(v_{\mathrm{S}}) = \mathrm{CPU}(v_{\mathrm{S}}) \cdot \frac{\sum_{j=1}^{i-1} \mathrm{link}(v_{\mathrm{V}}^j, v_{\mathrm{V}}^i)}{\sum_{j=1}^{i-1} \mathrm{hops}(M^V(v_{\mathrm{V}}^j), v_{\mathrm{S}}) \cdot \mathrm{link}(v_{\mathrm{V}}^j, v_{\mathrm{V}}^i)},$$

where $M^V(v_{\mathrm{V}}^j)$ refers to the substrate node that virtual node v_{V}^j is mapped to, $\mathrm{hops}(M^V(v_{\mathrm{V}}^j), v_{\mathrm{S}})$ refers to the substrate path with the fewest hops between substrate nodes $M^V(v_{\mathrm{V}}^j)$ and v_{S} in the SN, and $\sum_{j=1}^{i-1} \mathrm{link}(v_{\mathrm{V}}^j, v_{\mathrm{V}}^i)$ refers to the number of virtual nodes that have virtual links between the virtual node v_{V}^i in the set $v_{\mathrm{V}}^1, v_{\mathrm{V}}^2, ..., v_{\mathrm{V}}^{i-1}$. The value of $\mathrm{link}(v_{\mathrm{V}}^i, v_{\mathrm{V}}^j)$ is 1 when virtual nodes v_{V}^i and v_{V}^j are connected by a virtual link, and 0 otherwise.

NSIF considers the topology of the VN in the node mapping phase. The processes are as follows. When we map the ith node v_{V}^i according to the sorted sequence, we know the nodes $v_{\mathrm{V}}^1, v_{\mathrm{V}}^2, ..., v_{\mathrm{V}}^{i-1}$ have already been mapped to the SN. To reduce the average number of hops of substrate paths, we should make the two virtual nodes that have a virtual link between them not be mapped too far away in the SN. Thus, when we map v_{V}^i, we also consider whether the nodes $v_{\mathrm{V}}^1, v_{\mathrm{V}}^2, ..., v_{\mathrm{V}}^{i-1}$ have virtual links with v_{V}^i. If there is a virtual link between v_{V}^i and v_{V}^j ($0 \leq j \leq i-1$), we should select the substrate $M^V(v_{\mathrm{V}}^i)$ that is close to $M^V(v_{\mathrm{V}}^j)$.

Among the virtual nodes $v_{\mathrm{V}}^1, v_{\mathrm{V}}^2, ..., v_{\mathrm{V}}^{i-1}$, we need to select a substrate node v_{S} that is close to all the nodes $M^{\mathrm{V}}(v_{\mathrm{V}}^j)$ ($0 \leq j \leq i-1$), if there is a virtual link between v_{V}^i and v_{V}^j. Thus, we know that the substrate node should minimize

$$\frac{\sum_{j=1}^{i-1} \mathrm{link}(v_{\mathrm{V}}^j, v_{\mathrm{V}}^i)}{\sum_{j=1}^{i-1} \mathrm{hops}(M^V(v_{\mathrm{V}}^j), v_{\mathrm{S}}) \cdot \mathrm{link}(v_{\mathrm{V}}^j, v_{\mathrm{V}}^i)},$$

which represents the average number of hops of substrate paths. Meanwhile, considering the remaining CPU of the substrate node, we obtain the node selection factor NSIF.

Algorithm 1 Node mapping

for each VN request in order **do**
  Sort the virtual nodes according to their node degree DG_i in descending order
  for each node of the VN request in order **do**
    For the ith virtual node v_{V}^i, compute the NSIF values of the substrate nodes, and map them to the node with the largest NSIF value
  end for
  if node resource constraint is satisfied **then**
    return NODE_MAPPING_SUCCESS
  else
    return NODE_MAPPING_FAILED
end for

3.2 Link mapping phase

In the link mapping phase, similar to Yu *et al*. (2008) and Cheng *et al*. (2011), we adopt the *k*-shortest path algorithm (Eppstein, 1994) to map virtual links to substrate paths.

Algorithm 2 Link mapping

Input: The node mapping pairs generated by Algorithm 1.
Output: The substrate paths for node mapping pairs.
Map the virtual links using the *k*-shortest path algorithm
if link resource constraints are satisfied **then**
  return LINK_MAPPING_SUCCESS
else
  return LINK_MAPPING_FAILED
end if

3.3 Time complexity analysis

Suppose there are n VNs, the average number of nodes for each VN is p, the number of substrate nodes in SN is m, and the number of substrate links in SN is q. We know the complexity of the algorithm is polynomial-time.

The time complexity of sorting p virtual nodes is $O(p^2)$, and for n VNs, the time complexity is $O(np^2)$. The time complexity of Algorithm 1 is based on the second **for** loop. The time of computing the shortest path between two nodes is $O(m^2)$, and the time of computing NSIF is $O(m^2p)$. Thus, the time complexity of Algorithm 1 is $O(m^2p)\times O(n)=O(m^2pn)$. Algorithm 2 adopts the k-shortest path algorithm (Eppstein, 1994) to map the virtual links, and thus can be solved in polynomial time.

In summary, the time complexity of our mapping algorithm is max{$O(np^2)$, $O(m^2pn)$, polynomial}; thus, our algorithm can be solved in polynomial time.

4 Simulation results

In this section, we first introduce the performance evaluation environment, and then present our main evaluation results.

4.1 Evaluation settings

We have implemented a simulator to evaluate the performance of our algorithm. The simulator is a modified version of the VN mapping simulator devised by Princeton University, as mentioned in Yu *et al*. (2008).

Similar to Yu *et al*. (2008), in our experiment, the SN topology was configured to have 100 nodes with about 570 links, corresponding to a medium sized Internet service provider (ISP). We used the GT-ITM tool (Zegura *et al*., 1996) to generate the SN. The CPU and BW resources of the nodes and links were real numbers uniformly distributed between 50 and 100.

For each VN, the number of virtual nodes was determined by a uniform distribution between 2 and 20. The average VN connectivity was fixed at 50%, which means that an n-node VN has $n(n-1)/4$ links on average. The required CPU and BW of virtual nodes and links were real numbers uniformly distributed between 0 and 50. The arrivals of VN requests were modeled by a Possion process, with two requests per minute. The lifetime of the VN requests followed an exponential distribution with an average of 10 min. We ran each of our simulations for about 60 min.

We considered two scenarios, as the relative factor α was set to 1 and 2, respectively.

4.2 Evaluation results

Our evaluation results quantified the efficiency of the two algorithms. Several performance metrics for the evaluation purposes were used, including the long-term average revenue, acceptance ratio, average number of hops of the substrate paths, R/C ratio, and runtime of the algorithms. We also evaluated our results from three aspects, the general VNs and SN, the SN with changeable nodes, and the VNs with changeable required CPU or BW. We summarize the key observations from our simulation as follows.

4.2.1 General VNs and SN

The attributes of the general VNs and SN were the same as in the assumption, and would not change.

Figs. 4a and 4b show the R/C ratio of the two algorithms while setting α=1 and α=2, respectively. The R/C ratio of the topology awareness algorithm was obviously larger than that of the baseline algorithm.

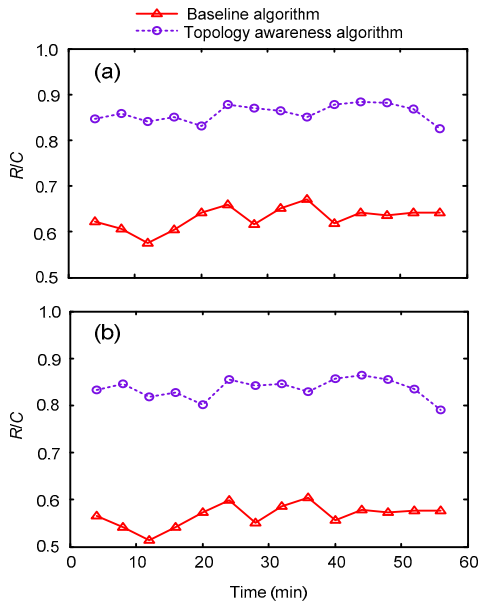


Fig. 4 R/C comparison between the baseline algorithm and the topology awareness algorithm with α=1 (a) and α=2 (b)

The reason is that the virtual links mapped by our algorithm are shorter than by the baseline algorithm. The former considers reducing the hops of the substrate paths by using the NSIF as the substrate node selection index in the node mapping phase, while the latter does not.

As shown in Fig. 5, the average number of hops of the topology awareness algorithm was obviously smaller than that of the baseline algorithm. For the topology awareness algorithm, the average number of hops was around 1.1, while for the baseline algorithm, it was around 2.0. According to Eq. (3), for the same VNs, the average cost of the latter was larger than that of the former (Fig. 4).

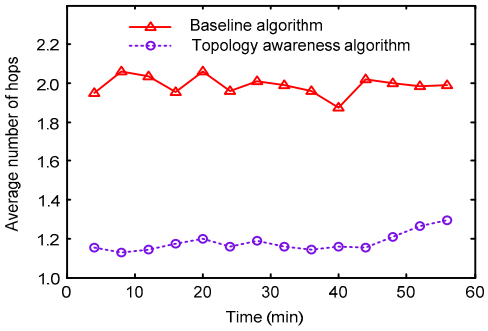


Fig. 5 Comparison of the average number of hops between the baseline algorithm and the topology awareness algorithm

We also compared the time complexity of the two algorithms. Without loss of generality, we compared the time complexity of mapping the same number of VNs to the SN.

Fig. 6 shows that the time complexity of the two algorithms was basically linear, and they were both polynomial time algorithms. The time complexity of the topology awareness algorithm was larger than that of the baseline algorithm, still polynomial time. The reason is that the former algorithm computes the NSIF value and spends much time in computing the distance among the substrate nodes, while the latter computes the $H(v_S)$ value, which is simple.

Fig. 7 shows that the acceptance ratio of the topology awareness algorithm was significantly larger than that of the baseline algorithm. The reason is that the higher R/C ratio indicates a higher resource utilization, which further results in accepting more VNs at certain infrastructure resources.

Figs. 8a and 8b show the average revenue of the two algorithms, setting α=1 and α=2, respectively. The topology awareness algorithm produced higher average revenue than the baseline algorithm. As mentioned above, the acceptance ratio of the former was higher, and thus more VNs can be accepted in the same time window, which results in the higher revenue.

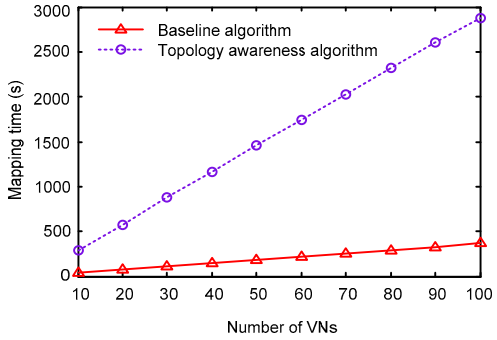


Fig. 6 Comparison of the runtime between the baseline algorithm and the topology awareness algorithm

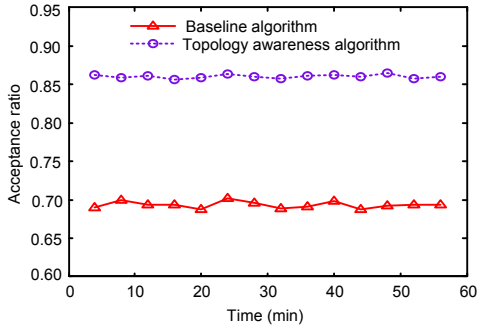


Fig. 7 Comparison of the acceptance ratio between the baseline algorithm and the topology awareness algorithm

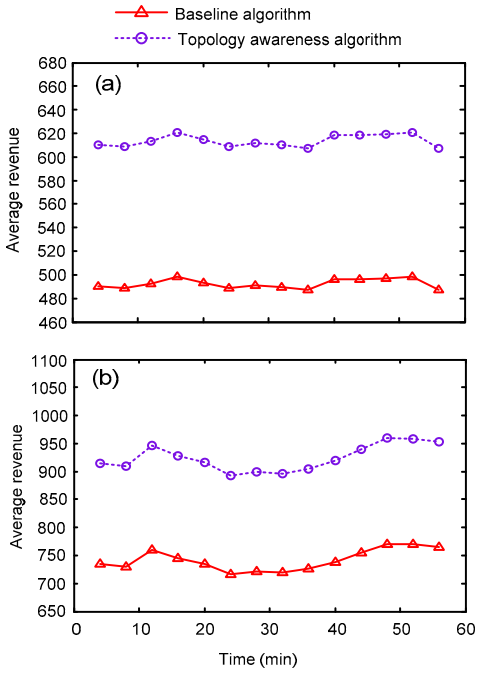


Fig. 8 Comparison of the average revenue between the baseline algorithm and the topology awareness algorithm with α=1 (a) and α=2 (b)

4.2.2 SN with changeable nodes

As the attributes of VNs are the same as in the assumption, we evaluated the maximum number of accepting VNs (the most VNs simultaneously running on the SN) as we changed the nodes of the SN.

Fig. 9 shows the maximum number of accepting VNs while setting the number of substrate nodes from 50 to 100. The results were approximately linear, and the result of the topology awareness algorithm was slightly better than that of the baseline algorithm.

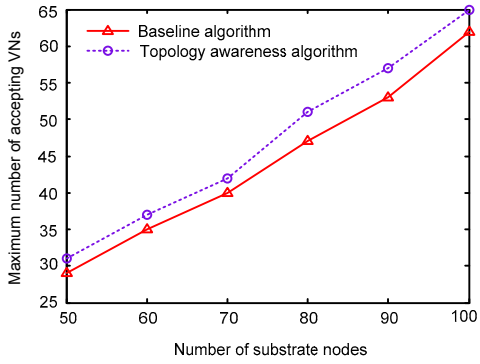


Fig. 9 Comparsion of the maximum number of accepting virtual networks between the baseline algorithm and the topology awareness algorithm

4.2.3 VNs with changeable required CPU or BW

In this experiment, we compared the *R*/*C* ratios of the two algorithms, as the requester's CPU or BW varied.

We assumed that the SN was fixed with 100 nodes and 570 links, and the topologies of VNs were also fixed. Now, we compared the *R*/*C* ratio with the required CPU of VNs increasing from 10% to 90% of the assumption (uniformly distributed between 0 and 50). The relative factor α was set to 1.

As shown in Fig. 10, the *R*/*C* ratio of the topology awareness algorithm was larger. Moreover, the *R*/*C* ratio became larger as the required CPU increased. This was more obvious for the baseline algorithm.

We also compared the *R*/*C* ratio with the required BW of VNs increasing from 10% to 90% of the assumption (uniformly distributed between 0 and 50). The relative factor α was set to 1.

As shown in Fig. 11, the *R*/*C* ratio of the topology awareness algorithm was larger than that of the baseline one. The *R*/*C* ratio became smaller as the required BW increased, and this was more obvious for the baseline algorithm.

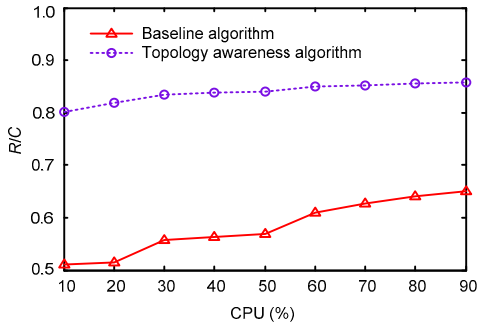


Fig. 10 The relationship between *R*/*C* and CPU for the baseline algorithm and the topology awareness algorithm (α=1)

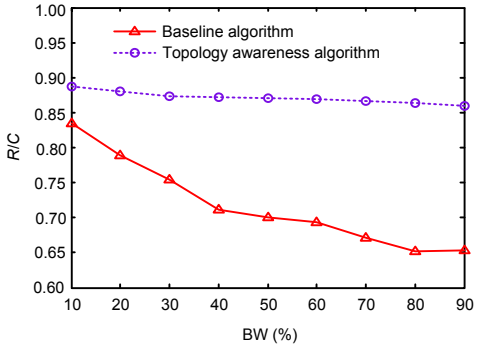


Fig. 11 The relationship between *R*/*C* and bandwidth (BW) for the baseline algorithm and the topology awareness algorithm (α=1)

5 Conclusions

In this paper, we present a new topology awareness virtual network mapping algorithm. The new algorithm is composed of two mapping phases. In the node mapping phase, it maps the virtual nodes to the substrate nodes. It selects the substrate node by NSIF, which considers the available CPU resource of the substrate nodes and the topology of the VN. In the link mapping phase, it maps the virtual links to the substrate paths using the *k*-shortest path algorithm.

We compared the performance of the new mapping algorithm with that of the baseline algorithm (Yu *et al*., 2008). Our findings are summarized as follows:

1. The new topology awareness algorithm produces a higher average revenue and acceptance ratio than the baseline algorithm.

2. The time complexity of the topology awareness algorithm is higher than that of the baseline algorithm, but it is still polynomial time.

3. The R/C ratio becomes larger as the virtual nodes' required CPU increases, and becomes smaller as the virtual links' required BW increases.

We will extend our work by considering dynamic change of the substrate network (e.g., substrate nodes or link failure, and topology change) and the specific application oriented VNMP (e.g., applications based on content delivery network (CDN), peer-to-peer (P2P), and cloud computing).

References

Anderson, T., Peterson, L., Shenker, S., Turner, J., 2005. Overcoming the Internet impasse through virtualization. *Computer*, **38**(4):34-41. [doi:10.1109/MC.2005.136]

Bansal, N., Lee, K.W., Nagarajan, V., Zafer, M., 2011. Minimum Congestion Mapping in a Cloud. Proc. 30th Annual ACM SIGACT-SIGOPS Symp. on Principles of Distributed Computing, p.267-276. [doi:10.1145/1993806.1993854]

Bavier, A., Feamster, N., Huang, M., Peterson, L., Rexford, J., 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.B., 2011. Virtual network embedding through topology-aware node ranking. *ACM SIGCOMM Comput. Commun. Rev.*, **41**(2):39-47. [doi:10.1145/1971162.1971168]

Chowdhury, N.M.M.K., Rahman, M.R., Boutaba, R., 2009a. Network virtualization: state of the art and research challenges. *IEEE Commun. Mag.*, **47**(7):20-26. [doi:10.1109/MCOM.2009.5183468]

Chowdhury, N.M.M.K., Rahman, M.R., Boutaba, R., 2009b. Virtual Network Embedding with Coordinated Node and Link Mapping. Proc. 28th IEEE Int. Conf. on Computer Communications, p.783-791. [doi:10.1109/INFOCOM.2009.5061987]

Eppstein, D., 1994. Finding the k shortest paths. *SIAM J. Comput.*, **28**(2):652-673. [doi:10.1137/S0097539795290477]

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]

Feamster, N., Gao, L., Rexford, J., 2007. How to lease the Internet in your spare time. *ACM SIGCOMM Comput. Commun. Rev.*, **37**(1):61-64. [doi:10.1145/1198255.1198265]

Guo, C., Lu, G., Wang, H.J., Yang, S., Kong, C., Sun, P., Wu, W., Zhang, Y., 2010. SecondNet: a Data Center Network Virtualization Architecture with Bandwidth Guarantees. Proc. 6th Int. Conf. on Emerging Networking Experiments and Technologies, p.15-26. [doi:10.1145/1921168.1921188]

Lischka, J., Karl, H., 2009. A Virtual Network Mapping Algorithm Based on Subgraph Isomorphism Detection. Proc. 1st ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures, p.81-88.

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

Turner, J.S., Taylor, D.E, 2005. Diversifying the Internet. IEEE Global Telecommunications Conf., p.755-760. [doi:10.1109/GLOCOM.2005.1577741]

Wang, Q.B., Jin, X., He, L., Zhao, Y., 2009. Virtualization and Cloud Computing. Publishing House of Electronic Industry, Beijing, China, p.26-30 (in Chinese).

Yu, M., Yi, Y., Rexford, J., Chiang, M., 2008. Rethinking virtual network embedding: substrate support for path splitting and migration. *ACM SIGCOMM 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. 15th IEEE Int. Conf. on Computer Communication, 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]