International Journal of Grid and Distributed Computing
Vol. 9, No. 9 (2016), pp.419-432
http://dx.doi.org/10.14257/ijgdc.2016.9.9.35

# Virtual Network Mapping Algorithm Based on Load Balancing

Ming Jiang [1], Xijie Tang [2], Min Zhang and Ziyang Li[4]

*1,2,3Institute of Software and Intelligent Technology, Hangzhou Dianzi University, Hangzhou 310018, China*
*4Hakim Information Technology Co., Ltd., Hangzhou 310018, China*
*[1]13588161992@163.com,[2]141050065@hdu.edu.com,[3]hz_andy@163.com,[4]liziyang @hakim.com.cn*

***Abstract***

*Recent studies for network virtualization have shown a promising way to overcome the Internet ossification. The one of the key issues in network virtualization is a virtual network mapping problem, i.e., mapping a virtual network to the physical network. The situations of dynamic arrivals of virtual network request and the limited life cycle of the virtual networks pose significant challenges to the virtual network mapping problem. A balance between the resource allocation of the physical network and the number of mapped virtual networks. In this paper, we have considered the time characteristics that virtual network requests when mapping algorithms so as to achieve the objective that the node load and link load can simultaneously reach a balance. Giving full consideration to mutual restraints of time and resources, we propose a two-dimensional discrete weighted model based on time and resources, and establish a mathematical programming model of minimizing the degree of two-dimensional load balancing. Moreover, we devise a VN embedding algorithms LB-VNE. Simulation experiments show that the proposed algorithms can increase the acceptance ratio and the revenue by the substrate network in the long term.*

***Keywords****: Network virtualization, time, virtual network embedding, Load balancing*

## 1. Introduction

Network Virtualization has recently received considerable attention in both academia and industry as an important enabler for designing the Future Internet architecture [1-2]. Network Virtualization technology will be a primary means of Internet innovation in the future, multiple separate virtual networks may build on the same underlying physical network, in order to achieve the goal of supporting simultaneously the network technology innovations and service upgrades. Network virtualization can effectively avoid the ossification of the Internet [3]. In a network virtualization environment, multiple service providers (SPs) will be able to create heterogeneous virtual networks (VNs) to offer customized end-to-end services to the end users by leasing shared resources from one or more infrastructure providers(InPs) without significant investment in physical infrastructure [4-5].

One of the important processes of network virtualization is to build a virtual network, the key issue of building virtual network is VN embedding which assigns substrate network resources to individual virtual network nodes and links. Specifically, each virtual node is assigned to one of the substrate nodes and each virtual link is assigned to a substrate path that connects the corresponding end nodes. The VN embedding problem is a NP-hard problem [5]. At present, there are two types of VN mapping algorithms. One is based on a heuristic algorithm that has appeared in the relevant literature [6-9]. Most of these studies focused primarily on edge mapping (using, for example, shortest path, k-shortest paths, and multi-commodity flow algorithms) after employing greedy methods to preselect the node sequence. The other is joint node and link mapping. A distributed collaborative algorithm

ISSN: 2005-4262 IJGDC

maps nodes and links with a VN to the substrate nodes and links simultaneously [10]. For example, the algorithm proposed in [11] uses a subgraph isomorphism detection to map the nodes and links. Due to the factor that VN requests are dynamic and unpredictable, in order to effectively utilize the physical network resources, most studies consider VN mapping to the substrate network with an objective of load balance [12-13].

Although these studies have their advantages, however, only a few studies considered the time issue [15-20], most of these studies did not consider the factor of a lifetime in the VN embedding. In reality, VNs arrive at different times and have different lifetime spans. On the other hand, a mapped virtual network occupies the physical network resources at run time. The resources are released at the end of the run time. The existing studies did not fully take into account the dynamic change of the physical network resources, which may lead to gradual deterioration of the utilization of the physical network resources over time.

To address the utilization deterioration problem, we studied the on-line VN embedding problem with specific focus on the time, aiming at the goal that the entire underlying physical network can reach equilibrium between the node load and link load at the same time. In this paper, we present a series of outcomes of this work, including the proposed two-dimensional discrete weighted model for time and resources, the values of the weighted metrics in the mapping algorithm, and the mathematical programming model that minimizes the degree of two dimensional load balancing. Our simulation results show that combined with VN mapping algorithm, substrate network resources can maintain balance stably in a long time.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 formalizes the network model and the VN embedding problem itself. In section 4, we establish load balancing virtual network embedding model based on time. Section 5 presents load balancing based virtual network embedding algorithm. Section 6 presents simulation results that evaluate the proposed algorithms and the paper concludes in Section 7.

## 2. Related Work

In this section, we discuss previous work in the area of virtual network embedding.

Zhu and Ammer *et al.* [6] proposed a set of four algorithms with the goal of balancing the load on the physical links and nodes, but their algorithms assumed that the physical network resources were unlimited, and the requests of multiple virtual networks arrived simultaneously. MinLan Yu *et al.*[7] proposed a greedy node mapping algorithm that prioritizes virtual networks with the largest revenue and link mapping using the shortest path with enough bandwidth capacity. If the request accepts path splitting, authors transformed the embedding problem into a multi-commodity flow problem. Besides, the goal was to achieve satisfied physical network resources utilization and load balancing. However, the above two literatures focused on the off-line problem, where all the requests were known in advance. It doesn't consider the time factor.

Chowdhury *et al.* [14] proposed VN embedding algorithms based on linear programming and rounding to solve the location-based embedded. Rost *et al.* [15] considered the virtual network that will not be static, it can over time and even include certain temporal flexibilities, and they proposed a continuous time mathematical programming approach to solve the temporal VN embedding problem.

Jiang *et al.* [16] considered the time attribute of the virtual network embedding, and presented a probability model which is formulated to obtain the maximum probability that the available resources of substrate network can be used by succeeding VN requests. However, their proposed algorithm that's based on a simple greedy embedding algorithm and had poor performance.

 

Our work is different from those early results. In contrast to the existed literatures, We study how to embed Virtual Network if the time factor is considered, and proposed a novel network mapping algorithm to achieve physical network resources load balancing.

## 3. Network Model and Problem Description

Solving the virtual network mapping problem involves graph models for the substrate network and the virtual network, most of these existing VN embedding studies hadn't considered the time issue. In this section, we first describe the network model of substrate network and virtual networks request that capture the dual nature of time and resources. Then we give the description of the load balancing problem in this new context.

### 3.1. Substrate Network Model

We model the topology of the substrate network as a weighted undirected graph $G_s = (N_s, E_s, A_s^n, A_s^e)$, where $N_s$ is the set of substrate nodes and $E_s$ is the set of substrate links between nodes of the set $N_s$. We use subscript to refer to substrate or virtual network, and use superscript to refer to nodes or links, unless otherwise specified. Substrate nodes and links are associated with their attributes, denoted by $A_s^n$ and $A_s^e$, respectively. And $A_s^n = \{A_{s(0)}^n, A_{s(1)}^n, A_{s(2)}^n, \cdots\cdots, A_{s(n)}^n, \cdots\}$,
$A_s^e = \{A_{s(0)}^e, A_{s(1)}^e, A_{s(2)}^e, \cdots\cdots, A_{s(m)}^e, \cdots\}$. Where $A_{s(n)}^n$ refers to CPU resource capacity of substrate node at time $n$, and $A_{s(m)}^e$ refers to bandwidth resource capacity of substrate link at time $m$. We also denote $P$ is the set of all loop-free paths in the substrate network. Figure 1 shows a substrate network, where the numbers over the links represent available bandwidths and the numbers in rectangles represent available CPU resources.

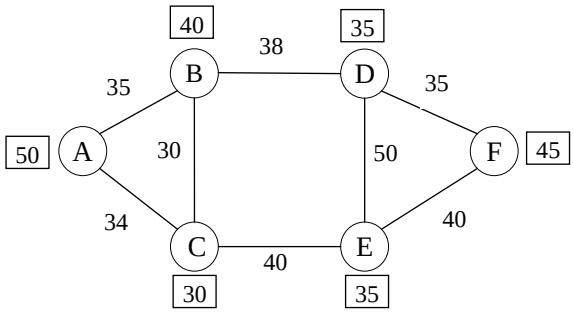


**Figure 1. Substrate Network**

### 3.2. Virtual Network Request

We model the topology of the virtual network as a weighted undirected graph $G_v = (N_v, E_v, A_v^n, A_v^e)$, where $N_v$ is the set of virtual nodes and $E_v$ is the set of virtual

links between nodes of set $N_v$. We denote by $A_v^n$ and $A_v^e$ the set of node CPU and link bandwidth constraints, respectively. Because each VN request has a running start time and duration, so the VN request is represented by the triple $VN = \{G_v, t_a, t_b\}$ ,where $t_a$ and $t_d$ represent the start-time and lifetime of the VN request, respectively. When the VN request arrives, if the request will accept, the substrate network allocates network resources on the substrate nodes and paths selected by that assignment for the VN Within the time constraints, and release the resources once the VN expires. When there are insufficient resources in the substrate network during the lifetime, the substrate network rejects the VN request. Figure. 2(a) and (b) shows two VN requests with node and link constraints.

### 3.3. VN Embedding Problem Description

A virtual network embedding for a VN request is defined as a mapping $M$ from $G_v$ to a subset of $G_s$, $M: G_V(N_V, E_v) \to G_s^{'}(N_s^{'}, P^{'})$, where $N_s^{'}$ is the subset of $N_s$, $P^{'}$ is the subset of $P$. The VN embedding can be decomposed into two major components: node mapping and link mapping.

Node mapping refers to mapping virtual nodes to the substrate nodes. It is defined by a mapping $M_{\mathrm{N}}: (N_v, A_v^n) \to (N_s^{'}, R_s^n)$, where $R_s^n$ is defined as the residual CPU capacity of the substrate node $R_s^n(n_s) = A_s^n(n_s) - \sum_{\forall n_v \to n_s} A_v^n(n_v)$, where $n_v \to n_s$ denotes that the virtual node $n_v$ is hosted on the substrate node $n_s$. The substrate node residual CPU capable supporting at least one virtual node request from time $t_a$ to time $t_a + t_d$, *i.e.* $A_v^n(n_v) \le R_{s(t)}^n(M_N(n_v)), t = t_a, \cdots, t_a + t_d$. In this paper, the virtual node of different virtual network request can be mapped to the same substrate node, but each virtual node from the same VN request must be assigned to a different substrate node.

Link mapping refers to each virtual link is assigned to a substrate path between the corresponding substrate nodes. It is defined by a mapping $M_e: (E_v, A_v^e) \to (P', R_s^e)$, where $R_s^e$ is defined as the residual bandwidth capacity of the substrate link. $R_s^e(e_s) = A_s^e(e_s) - \sum_{\forall e_v \to e_s} A_v^e(e_v)$, the substrate path residual bandwidth capable supporting at least one virtual link request from time $t_a$ to time $t_a + t_d$, *i.e.* $A_v^e(\mathrm{e}_v) \le R_{s(t)}^e(M_e(e_v))$, $t = t_a + t_d$.

Figure 2.(c) shows the VN embedding solutions for the two VN requests. For example, in Figure.2, the VN request 1 has the node mapping $\{a \to A, b \to B\}$, it has been assigned the link mapping $\{(a,b) \to (A,B)\}$. And the VN request 2 has the node mapping $\{c \to D, d \to C, f \to F\}$, it has been assigned the link mapping $\{(c,d) \to (D,B,C), (c,f) \to (C,E,F)\}$.

 

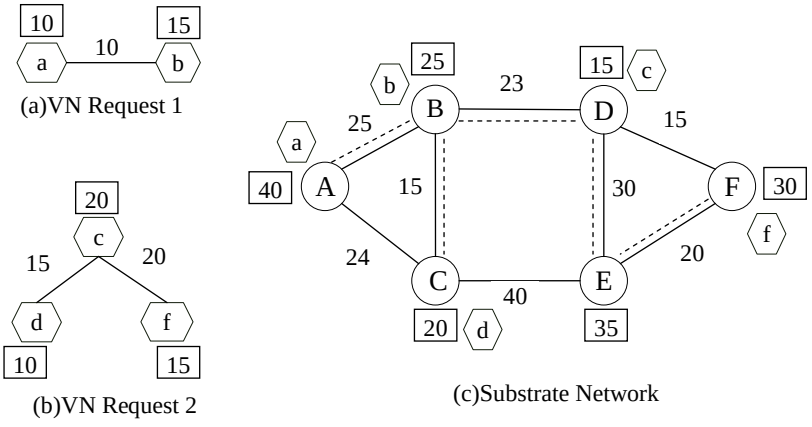


**Figure 2. An Example of VN Embedding**

We should consider the time factor when designing the virtual network mapping problems, because the VN request has a lifetime and start-time. We can know the importance of the time factor from the example as below.

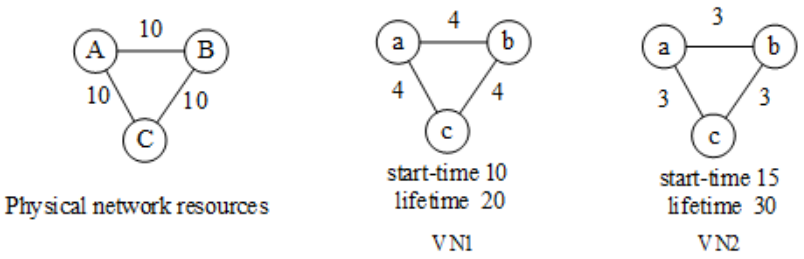


**Figure 3. Virtual Network Request**

From the Figure 3., there are two virtual networks have been mapped node, and the physical network would accept two virtual networks. Now the physical network will reject this request3 when VN3 arrived (Figure 4). Because the physical network has pre-allocated link resources for VN1 and VN2, there are no enough resources to meet VN3. In fact, the request 3 is acceptable, because VN1 finished mapping when the VN3 arrived, and the physical network has 7 units resources that could accept the VN3. Obviously, the physical network performance will be getting worse if we don't consider the time factor.

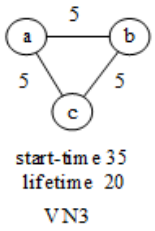


**Figure 4. Virtual Network Request 3**

### 3.4. Objectives

Our main interest in this paper is to propose an efficient online VN embedding algorithm where VN requests arrive and depart for time. We want to increase revenue in the long term, and in addition to balancing load of the substrate network resources. Similar to the previous work in [7], [14], we introduce some definitions as follows.

Definition 1 VN request acceptance ratio:

$$a = \lim_{T\to\infty} \frac{\sum_{t=0}^{T} m}{\sum_{t=0}^{T} n} \tag{1}$$

Where $\sum_{t=0}^{T} m$ refers to the number of successful mapping from time $t=0$ to time $T$, Where $\sum_{t=0}^{T} n$ refers to the total number of VN request from time $t=0$ to time $T$.

Definition 2 VN request revenue:

$$R(t) = \sum_{i=0}^{i=k} \alpha_i \left( \sum_{n\in n_v} cpu(n,t) + \sum_{e\in E_v} bw(e,t) \right) \tag{2}$$

Where $k$ refers to the number of VN request in time $t$, $\alpha_i \in \{0,1\}$, $\alpha_i = 1$ refers to the $i'th$ VN mapping successfully, $\alpha_i = 0$ refers to fails. $R(t)$ refers to the revenue of VN request at time $t$. $cpu(n,t)$ and $bw(e,t)$ are the CPU and bandwidth requirements for the virtual node $n$ and the virtual link $e$ at time t, respectively.

## 4. Load Balancing Model for Real-time Virtual Network Embedding

Because of the situation that physical network doesn't support path spilt, in this paper, we have considered the time characteristics to achieve the objective that network reach a balance. We establish a load balancing model for real-time virtual network embedding. Firstly, we should have some definition of the model, and the details as follows.

### 4.1. The Definition for Node Intensity and Link Intensity

Build a virtual network not only need to meet the demands of service providers, but also need to take efficient use the substrate network resources. It requests that the network resource allocation should be reasonable, reduce link congestion, and avoid failure in building a virtual network for some links congestion. Therefore we should build a virtual network on the principle of keeping network load balance.

Definition 3: node load strength $S_n(t)$:

$$S_n(t) = \frac{A_s^n - R_s^n(t)}{A_s^n} \tag{3}$$

Where $S_n(t)$ represents node load strength at $t$ time.

Definition 4: link load strength $S_e(t)$:

$$S_e(t) = \frac{A_s^e - R_s^e(t)}{A_s^e} \tag{4}$$

 

Where $S_e(t)$ represents link load strength at $t$ time. That is the ratio of the total bandwidth of the link bandwidth and the physical link used.

### 4.2. Two-dimension Load Balancing Model for Resources and Time Factor

The definition of the node and the link load strength above just considering the current loading conditions of substrate network status. If a virtual network is built based on current time, load balancing status of the entire physical network over time will gradually deteriorate. Because when the VN request arrives, it is dynamical and unpredictable. The dynamic Virtual network mapping and deleting will have an impact on the situation of the balance of physical network load, which may cause node overloading or locally blocked link. Based on this situation, in this paper, we will present the two-dimensional discrete weighted model to measure the nodes and links resource load conditions over time.

The time of virtual network effects load balancing of the substrate network. And within the time range, the loads on the link and node are constantly changing within different virtual networks start and end time. As the network service providers can't accept VN request for any a short period, the minimum time required can be discreted as unit that we can build a virtual network. Therefore, from the current time point of view, the load of each node and link is fixed. It is only considered that the resource property for load balancing, obtains load intensity within each time slice. Load intensity of different time slice, have different influence on the two-dimension load strength at the present time. The closer it is from current time, the greater the impact is. This is because we have just started to pre-allocate VN resources, if the time it needs to transfer the pre-allocated resources is closer to the current time, the price to transfer will be higher.

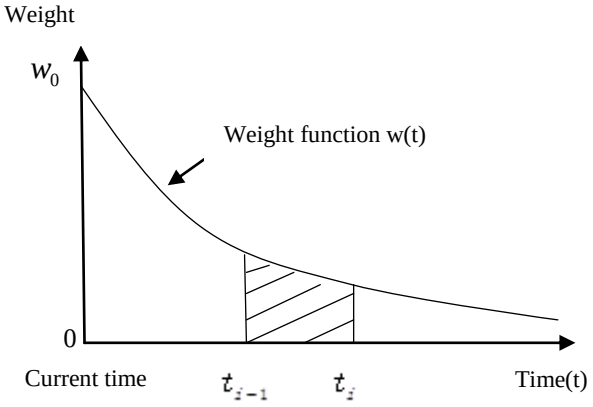


**Figure 5. Weight Function Schematic**

In view of this, as shown in Figure 5, we can introduce a monotonically decreasing function $w(t)$, and the function satisfies $\int_0^\infty w(t) = 1$. The start and end time of each time slice as integral upper and lower limits, respectively.

Definition 5: node two-dimensional load strength:

$$w_n = \sum_{i=1}^{\infty} (S_n(t_i) * \int_{t_i}^{t_{i+1}} w(t)dt) \tag{5}$$

Definition 6: link two-dimensional load strength:

$$w_e = \sum_{i=1}^{\infty} (S_e(t_i) * \int_{t_i}^{t_{i+1}} w(t)dt) \tag{6}$$

Definition 7: node two-dimensional load balance degree:

$$N_\sigma = \sqrt{\frac{\sum_{|N_s|}(W_n(N_s) - N_{avg})^2}{|N_s|}} \qquad (7)$$

$$N_{avg} = \sum_{N_s} W_n(N_s) / |N_s|$$

Where , $N_\sigma$ represents a standard deviation that physical network nodes two-dimensional load strength.

Definition 8: link two-dimensional load balance degree:

$$L_\sigma = \sqrt{\frac{\sum_{|E_s|}(W_e(E_s) - L_{avg})^2}{|E_s|}} \qquad (8)$$

$$L_{avg} = \sum_{|E_s|} W_e(E_s) / |E_s|$$

Where , $L_\sigma$ represents a standard deviation that physical network link two-dimensional load strength.

Remarks:

(1) $t_{i+1} - t_i$ represents the smallest unit of time, the minimum time unit that network service providers need to provide services for user.

(2) $w(t)$ is a weight function, it must satisfy $\int_0^\infty w(t) = 1$, and strictly monotonically decreasing, infinitely approach to a threshold. This paper takes the Gaussian distribution function $w(t) = \int_0^{+\infty} \frac{2}{\sqrt{\pi}} e^{-t^2} dt$ that subject to the conditions required.

(3)Formula(5),(6) expresses the starting and ending time of each time slice as integral upper and lower limits respectively, integral weight function to get weights of load balancing about the time slice, and the load of the time slice on a weighted sum. In order to obtain a two-dimensional load balancing degree in the virtual network operation start and end time.

### 4.3. Minimize the Degree of Two-dimensional Load Balancing Mathematical Programming Model

In order to achieve the best load balancing in the substrate network resources, we define the physical network load balancing degree: $u = \alpha N_\sigma + \beta L_\sigma$, the objective function is:

$$Minimize \quad u = \alpha N_\sigma + \beta L_\sigma \qquad (9)$$

Constraints:

$$A_v^n(n_v) \le R_{s(t)}^n(M_N(n_v)), t = t_a, \cdots, t_a + t_d \qquad (10)$$

$$A_v^e(e_v) \le R_{s(t)}^e(M_e(e_v)), t = t_a, \cdots, t_a + t_d \qquad (11)$$

$\alpha$ and $\beta$ are dynamic adaptive adjustment within the range, according to the node and link load balancing based on the status of the entire physical network after a VN request has been mapped. When load balancing degree of the link or node is poor, increase its weighting factor to improve load balancing of link or node. Therefore, it is can maintain better balanced load among all substrate links and substrate nodes.

 

## 5. Real-time Virtual Network Embedding Algorithm for Load Balancing

Since solving above mathematical programming model is known to be NP-hard problem, it is very difficult to get the global optimal solution that directly used program approaches, and heuristics will be used to solve the problem. The existing heuristics,such as Backtracking algorithm, Simulated Annealing, Approximation Algorithms, these algorithms request complex calculation and take higher cost, and they often not give more reasonable and effective mapping of sub-optimal solution. So most of these heuristics first map node with greedy algorithm and then focus on the link mapping. The heuristic algorithm is more simple and effective, but that has no pre-selection node in the link mapping stage, and it also does not consider the use efficiency of the physical network resources.

In this paper, we devise virtual network mapping algorithm of load balancing (LB-VNE). The algorithm is divided into two processes. First of all, seeking an initial feasible solution (step1 to step4) for the dynamic arrival of virtual network, the virtual network is mapped onto the substrate network that has smaller two-dimensional load strength. Secondly, Iterative optimization initial feasible solution (step5 and step6): calculated physics network load balance degree of current feasible solution. Successively changing the virtual node corresponding physical node until all virtual nodes complete, if physics network load balance degree can be reduced, and accept the changes.

Since VN request arrived at a certain time, when the received request can be mapped immediately. If it exceeds the running start time, the virtual network request fails. Therefore we analyzed each virtual network mapping algorithm. The algorithm is as follows:

**Algorithm** VN embedding algorithm(LB-VNE)

Step 1 Sort the virtual nodes according to CPU demands in descending order.
Step 2 Mapping all virtual nodes sequentially
  Step 2.1 Find the subset $S$ of substrate nodes that satisfy available CPU capacity for the virtual node and not marked. If $S$ is empty, and refuse the VN request.
  Step 2.2 For the virtual node, find the substrate node for $S$ with the minimum $w_n$ (defined in Equation (5)). Then mark the substrate node and go to step 2.1.
Step 3 Sort the virtual links according to bandwidth demands in descending order.
Step 4 Mapping all virtual links sequentially.
  Step 4.1 Find the set $L$ of k-shortest paths (k is 4) that satisfy available bandwidth capacity for the virtual link. If $L$ is empty, and refuse the VN request.
  Step 4.2 In the set $L$, find a path with the minimum $w_e$ (defined in Equation (6)). Then go to step 4.1.
Step 5 Calculate physics network load balance degree $u$ (defined in Equation (9)) for the current mapping scheme
Step 6 In the current mapping scheme, each virtual node complete successively:
  Step 6.1 Find the subset $M$ of substrate nodes that satisfy available CPU capacity for the virtual node (since some of the resources of the substrate network has been occupied by VN according to the current mapping scheme, it is not necessarily the same as set of $M$ ).
  Step 6.2 Modify the current VN embedding scheme, replace the current physical node that the virtual node has mapped with the substrate node of the set $M$ , and corresponding to the reduced value $u$ (before calculating $u$ , remapping the current virtual node associated links by Step4.1 and Step4.2), select the substrate node instead of the previous physical nodes.

## 6. Performance Evaluation

In this section, we firstly describe the evaluation environment, and then present our main evaluation results.

### 6.1 Simulation Settings

We have implemented a discrete event simulator to evaluate our embedding algorithm. The substrate network topology is generated by the GT-ITM tool [21], the substrate network contains 30 nodes and 80 links, each pair of substrate nodes is randomly connected with probability 0.5. The CPU and bandwidth resources of the substrate nodes and links are real numbers uniformly distributed between 50 and 100. We assume that VN requests arrive and start running time obey Poisson distribution with $\lambda$ = 25 time units, and each one has an exponentially distributed lifetime with an average of $\lambda$ = 100 time units. In each VN request, the number of virtual nodes is randomly determined by a uniform distribution between 2 and 5. Each pair of virtual nodes is randomly connected with probability 0.5. The CPU and bandwidth resources of the virtual nodes and links are real numbers uniformly distributed between 0 and 50.

### 6.2 Evaluation Results

To better illustrate the advantages of LB-VNE, in this paper, we compared with research [16] measure mentioned algorithm(show as Table 1). We use several performance metrics for evaluation purposes in our experiments. We measured the VN request acceptance ratio, revenue, and the network load balance.

**Table 1. Compared Algorithms**

| Notation | Algorithm description |
|---|---|
| LB-VNE | Based on time and source discrete-weighted load balancing of VN mapping algorithm. |
| ML_VNE | Using Maximum load value of remaining physical network resources as a measure |

The simulation results are shown in Figure 6 to Figure 8.

Figure 6 depicts that algorithm LB-VNE leads to better acceptance ratio than the ML-VNE. This is because the VN mapping with LB-VNE algorithm, take full account of the impact of physical network load intensity at different times for the present time physical network load balancing, which allows the physical network to maintain a better state of load balancing in the long term. The ML-VNE algorithm is only a simple judgment of the current physical network resources, and doesn't consider the impact on the future of physical network balancing.

From Figure 7, it is evident that LB-VNE generates more revenue than ML-VNE algorithm. Due to the LB-VNE fully take into consideration resource load interaction at each time slice, from which VN mapping strategy benefits for a long term. ML-VNE simply considers the maximum network resource load during a period of time. Therefore LB-VNE has better performance in physical network load balance, it could accept more VN request to obtain more benefits under the same conditions of physical resources.

From Figure 8, we could find that LB-VNE has lower network load balancing degree, that indicates better uniform distribution of physical network resources.

 

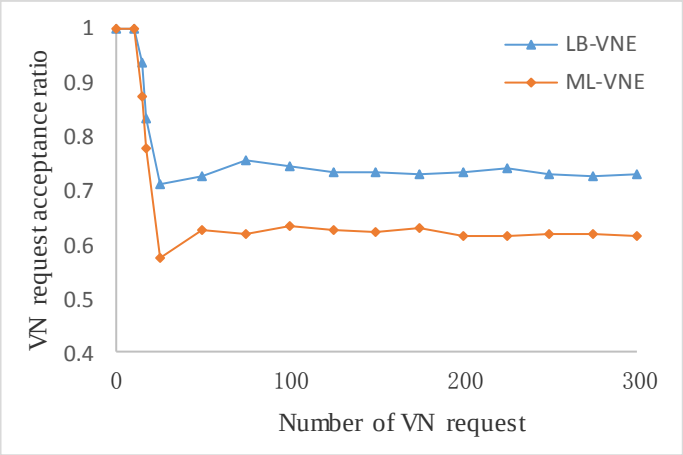


**Figure 6. VN Request Acceptance Ratio**

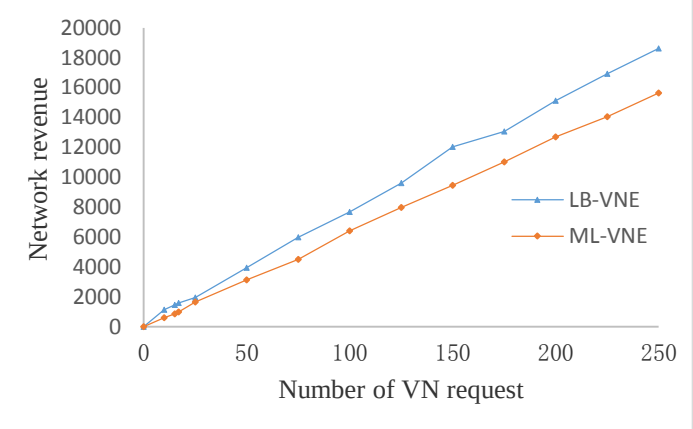


**Figure 7. Network Revenue**

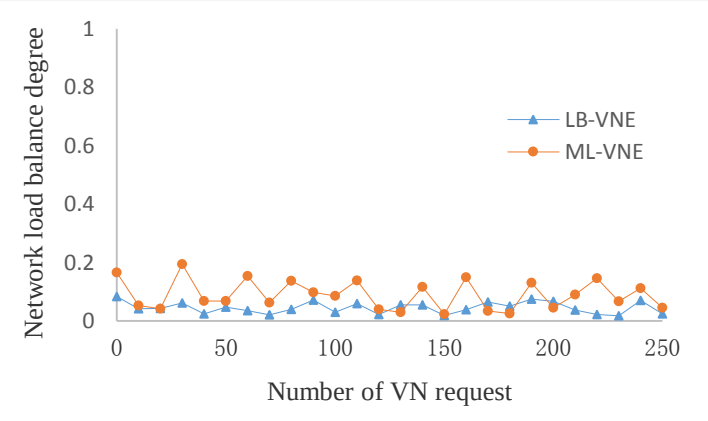


**Figure 8. Physical Network Load Balance Degree**

## 7. Conclusion

Network virtualization is a promising way to de-ossify the current Internet by providing a shared platform for a variety of different network services and architectures. In this paper, we discussed the two-dimensional property of time and space, we developed a basic scheme

for VN embedding based on load balancing. Iterative heuristic strategy is presented to further improve the performance. Simulation experiments show that the LB-VNE algorithms increase the acceptance ratio and the revenue through the substrate network in the long term.

## Acknowledgments

This work is supported by 863 Program (2015AA011901、2015AA015602) and Zhejiang Province Science and technology project (No. 2014C01051) and Zhejiang Province Key scientific and technological innovation team (2013TD20).

## References

[1] Global Environment for Network Innovations (GENI), http://www.geni.net
[2] Future Internet Network Design (FIND), http://find.isi.edu
[3] T. Anderson, L. Peterson, S. Shenker and J. Turner, "Overcoming the Internet impasse through virtualization", Computer, vol. 38, no. 4 **(2005)**, pp. 34-41.
[4] A. Bavier, N. Feamster, M. Huang, L. Peterson and J. Rexford, "In VINI veritas: realistic and controlled network experimentation", Acm Sigcomm Computer Communication Review, vol. 36, no. 4 **(2010)**, pp. 3-14.
[5] N. Feamster, L. Gao and J. Rexford, "How to lease the internet in your spare time", Acm Sigcomm Computer Communication Review, vol. 37, no. 1 **(2007)**, pp. 61-4.
[6] Y. Zhu and M. Ammar, "Algorithms for Assigning Substrate Network Resources to Virtual Network Components", Infocom IEEE International Conference on Computer Communications, **(2006)**, pp. 1-12.
[7] M. Yu, Y. Yi, J. Rexford and M. Chiang, "Rethinking virtual network embedding: Substrate support for path splitting and migration", Acm Sigcomm Computer Communication Review, vol. 38, no. 2 **(2015)**, pp. 17-29.
[8] J. Lu and J. Turner, "Efficient Mapping of Virtual Networks onto a Shared Substrate", Washington University in St Louis, **(2006)**.
[9] R. Ricci, C. Alfeld and J. Lepreau, "A Solver for the Network Testbed Mapping Problem", Acm Sigcomm Computer Communication Review, vol. 33, no. 2 **(2010)**, p. 2003.
[10] I. Houidi, W. Louati and D. Zeghlache, "A Distributed Virtual Network Mapping Algorithm", IEEE International Conference on Communications, **(2008)**, pp. 5634-40.
[11] J. Lischka and H. Karl, "A virtual network mapping algorithm based on subgraph isomorphism detection", ACM Workshop on Virtualized Infrastructure Systems and Architectures, **(2010)**, pp. 81-8.
[12] Q. Ning, "Research on Balanced Construction Algorithm of Virtual Network", Dianzi Yu Xinxi Xuebao/journal of Electronics & Information Technology, vol. 33, no. 6 **(2011)**, pp. 1301-6.
[13] Wang H X, Jiang M, Fu J., "Research on load-balanced construction algorithm of logical carrying network." Journal on Communications **(2012).**
[14] N. M. M. K. Chowdhury, M. R. Rahman and R. Boutaba, "Virtual Network Embedding with Coordinated Node and Link Mapping", Proceedings - IEEE INFOCOM, vol. 20, no. 1 **(2009)**, pp. 783-91.
[15] M. Rost, S. Schmid and A. Feldmann, "It's About Time: On Optimal Virtual Network Embeddings under Temporal Flexibilities", IEEE International Parallel & Distributed Processing Symposium, **(2014)**, pp. 17-26.
[16] JIANG, Ming, ZHAO, Zhiyang, ZHANG, TANG, Jingfan, Chunming and Xiao, "A Virtual Network Mapping Algorithm Based on Time", Chinese Journal of Electronics, vol. 23, no. 1 **(2014)**, pp. 31-6.
[17] T. Huang, Y. Gu, J. Liu and Y. Liu, "Time Efficient Virtual Network Embedding Algorithm", Intelligent Automation & Soft Computing, **(2016)**, pp. 1-8.
[18] M. Bui, T. Wang, B. Jaumard and D. Medhi, "Time-varying resilient virtual network mapping for multi-location cloud data centers", International Conference on Transparent Optical Networks, **(2014)**, pp. 1-8.
[19] S. Zhang, Z. Qian, S. Guo and S. Lu, "FELL: A Flexible Virtual Network Embedding Algorithm with Guaranteed Load Balancing", Communications (ICC), 2011 IEEE International Conference on, **(2011)**, pp. 1-5.
[20] S. Zhang, Z. Qian, J. Wu, S. Lu and L. Epstein, "Virtual Network Embedding with Opportunistic Resource Sharing", IEEE Transactions on Parallel & Distributed Systems, vol. 25, no. 3 **(2014)**, pp. 816-27.
[21] E. W. Zegura, K. L. Calvert and S. B. Acharjee, "How to model an internetwork", Fifteenth Joint Conference of the IEEE Computer & Communications Societies Conference on the Conference on Computer Communications, **(1996)**, p. 594.

 

## Authors

**Ming Jiang**, received the Ph.D. degree in computer science from Zhejiang University in 2004, and currently is a professor of College of Computer Science at Hangzhou Dianzi University. His research areas include network virtualization, software defined network and Internet QoS provisioning.

**Xijie Tang**, currently is a postgraduate of College of Computer Science at Hangzhou Dianzi University. His research areas include network virtualization, software defined network.

**Min Zhang**, received the Ph.D. degree in computer science from Zhejiang University in 2012, and currently works at College of Computer Science at Hangzhou Dianzi University. His research areas include network virtualization, software defined network and Internet QoS provisioning.

**Ziyang Li**, received bachelor's degree from Zhejiang University of Technology in 2008, and currently is the technical director for Hakim Information Technology Co., Ltd, mainly engaged in cloud data center product research and development.

 