A Virtual Network Embedding Algorithm Based on Graph Theory

Zhenxi Sun, Yuebin Bai, Songyang Wang, Yang Cao, Shubin Xu

A Virtual Network Embedding Algorithm based on Graph Theory

Zhenxi Sun[1], Yuebin Bai[1,2,*], Songyang Wang[1], Yang Cao[3], and Shubin Xu[2]

[1] School of Computer Science and Engineering, Beihang University, Beijing 100191, China
*yuebinb@gmail.com
[2] Science and Technology on Information Transmission and Dissemination in Communication Networks Laboratory, Shijiazhuang 050081, China
[3] School of Computer Science,Beijing University of Posts and Telecommunications, Beijing 100876, China

Abstract. Network virtualization is becoming a promising way of removing the inherent ossification of the Internet, and has been steadily attracting more and more researchers' attention during the decades. The major challenges in this field are improving the efficiency of virtual network embedding procedure and raising the rate of virtual network requests being successfully accepted by the substrate network. This paper introduces a new virtual network embedding algorithm based on node similarity, which means the similarity between the virtual nodes and the substrate nodes. For more details, by calculating the degree of nodes both in virtual network and substrate network, which is actually the number of links associated with them, the algorithm achieves better mapping results between virtual network and the substrate network on the topology aspect.

Keywords: Virtual Network; VN Embedding; Graph Theory; Node Similarity

1 Introduction

For recent years, Internet is not only greatly changing the ways that people communicating with each other, but also making a profound influence on the whole society. During the past four decades, the internet architecture has proven its great worth by meeting most of the requirements of distributed applications, which is improving the life of whole world. However, the remaining problems and defects of the current internet are becoming more and more prominent, such as extendibility, manageability, quality of service and power-saving, and so on. In order to solve these problems fundamentally, some researchers propose an idea of designing the architecture of future network from clean-slate so that to break the constrains of current internet. However, no matter how to change the architecture of internet, what must be kept in mind are the following three requirements: firstly, researchers must be able to experiment easily with new architectures

on live traffic from the existing networks. Secondly, there must be a plausible deployment path for putting validated architectural ideas into practice. Thirdly, the proposed solutions should be comprehensive so that they can address the broad range of current architectural problems facing the Internet [1], instead of focusing on a single narrow problem.

To keep up with such requirements, network virtualization shows the elegant charm and powerful vitality. It allows users create virtual networks which can be viewed as the normal networks used in real life and work, then maps them to the substrate networks which can be the networks with traditional internet architectures or other already deployed networks. In this way, the users can use the virtual network to meet their demands, for example, trying the new network architecture or providing services for companies or single users. By enabling a plurality of diverse network architectures to coexist on a shared physical substrate network, virtualization mitigates the ossifying forces at work in the current internet and enables continual introduction of innovative network technologies. Such a diversified Internet would allow existing architectural deficiencies to be holistically addressed as well as enable the introduction of new architectures supporting new types of applications and services [2].

Currently, virtual network embedding algorithms mainly parted in two periods: *virtual node mapping period* and *virtual link mapping period* [3]. What's more, virtual network embedding problem, with constrains on virtual nodes and virtual links, can be reduced to the *NP*-hard *multi-way separator problem* [3], even if all the request are known in advance. Even when all the virtual nodes are already mapped, embedding the virtual links with bandwidth constraints onto substrate paths is still *NP*-hard in the *indivisible flow scenario*. As a result, there are many heuristic algorithms appears in this research area [4–7]. Although the efforts on network virtualization research view the problem in case of the fixed networks or stable networks, the efficiency on mapping virtual networks to the substrate networks still has a huge improving space. What's worse, the mobile substrate networks are paid little attentions. What will happen if the substrate network is an unstable network environment, such as MANET (Mobile Ad hoc Networks).

In order to address the network virtualization of such environments, we devise a novel mapping algorithms with forecasting techniques in this paper. Firstly, we divide the nodes in physical networks into several small groups by the state of link between nodes, then map the virtual nodes to nodes of physical networks. Secondly, after all the virtual nodes have been mapped to physical nodes, the virtual link mapping phase is finished with multi-commodity flow algorithms.

The rest of the paper is organized as follows. Section II shows the related work in network virtualization field. Section III formulates and models the virtual network embedding problem. Section IV introduces the virtual network embedding algorithms. With the experiments result of the whole algorithm analyzed in section V, the paper will draw a conclusion in section VI.

2 Related Works

Finding the optimal VN mapping solution that satisfies multiple objectives and constraints can be formulated as an *NP*-hard problem. The aspects of VN mapping algorithms appears until now include independent virtual node mapping and virtual link mapping, introduce better correlations between the two phases by facilitating the latter phase when mapping the virtual nodes to substrate networks, and simultaneously mapping virtual nodes and virtual links to the physical networks.

Being different from the embedding algorithms [4–7] that separate the virtual node mapping procedure and the virtual link mapping procedure, authors in [8, 9] introduces better correlation between the node mapping and the link mapping phases by proposing two new VN embedding algorithms D-ViNE (Deterministic VN Embedding) and R-ViNE (Randomized VN Embedding). The VN embedding problem deals with the mapping of a VN request, with constraints on virtual nodes and links, onto specific physical nodes and paths in the substrate network that has finite resources. Since multiple VNs share the underlying physical resources, efficient and effective embedding of online VN requests is of utmost importance in order to increase the utilization of substrate network resources and InP revenue [8]. Authors in [8] use mixed integer programming (MIP) formulation [10]to solve the embedding problem with binary constraints on the meta edges and linear constraints on actual substrate network links. Once all the virtual nodes have been mapped, they use the multi-commodity flow algorithm to map the virtual links onto substrate network paths between the mapped virtual nodes [4]. While the algorithms are not taking the changing topology of the substrate network into account, [11] introduces a topology-aware note ranking method. Inspired by Google's PageRank measure, authors devise a Markov random walk model, for computing topology-aware resource ranking of a node to reflect both the resource and the quality of connections of the node in the network.

The authors in [12] have proposed a distributed algorithm that simultaneously maps virtual nodes and virtual links without any centralized controller, but scalability and performance of their algorithm is still not comparable with the centralized ones. Another distributed and autonomic virtual network mapping framework is proposed in [5], where the substrate nodes integrates autonomous and intelligent agents which exchange messages and cooperate to carry out the proposed VN mapping algorithm. The paper takes nodes integration into account, but not estimate the influence from node movements brings to VN mapping efficiency. Although these algorithms have been verifying the mapping procedure in wireless network environments, physical node mobility of wireless network in the physical network environment is not taken into consideration.

3 Network Model and Problem Description

For clearly describing the VN embedding problem in MANET environments, the notation of the key elements should be defined in the very starting point. On

the one hand, the notations can facilitate describing VN embedding algorithms; on the other hand, they will show what aspects the solutions have taken into consideration.

3.1 Substrate Network Description

We model substrate network an undirected and weighted graph, and note it as $G^s = (N^s, E^s)$, where G^s denotes the substrate network, N^s denotes the set of nodes in substrate network and E^s denotes the set of substrate links. Each node of substrate network $n^s \in N^s$ is associated with CPU capacity $c(n^s)$ and location information $loc(n^s)$. And each link of substrate network $e^s \in E^s$ is associated with the bandwidth $b(e^s)$, where $e^s(i, j)$ denotes the bandwidth from the node i to node j.

What the most obvious difference between MANET and traditional internet is that the influence of node mobility must be taken into consideration in the former environments. Due to the movements of substrate nodes, the substrate links would be disrupted frequently. And the topology of the substrate network finds itself in a dynamically changing state. The initial reason is the mobility of the substrate nodes, while the ultimate result is the connected and disconnected states switching in substrate links. In order to applied network virtualization into such circumstance, the link state must be taken into consideration. Thus, we define the stability of the substrate link e^s as $s(e^s) \in [0, 1]$, where the bigger value represents the more stable state of e^s.

3.2 Resources of Substrate Network

Each virtual network request should meet the following requirements. The computation requirements of virtual nodes is ultimately the computing speed of the CPU. When the virtual nodes need a 1Ghz computing speed, then the physical node should be 1GHz at least, and the higher the better. So the first requirement must be satisified is $c(n^v) \leq c(n^s)$. Similarly, the bandwidth requestment of virtual link should be smaller than the mapped physical links of substrate network. In another word, when mapping the virtual link to phsical links, the physical links bandwidth should be greater than that of the virtual link. Thus, the second requirements must be meet is $b(n^v) \leq b(n^s)$. While, the above two requirements should be applied when the physical nodes have not been mapped any virtual nodes. If not so, we should consider the remaining resources of the physical nodes and links of substrate network. The remaining computing resources (i. e. CPU capacity) is denoted by

$$\mathcal{R}_N(n^s) = c(n^s) - \sum_{\forall n^v \uparrow n^s} c(n^v)$$

which means that the substrate computing resources of the substrate node n^s is the total computing resouces subtract the resouces that already allocated to

virtual nodes. In the same way, the substrate bandwidth is denoted by

$$\mathcal{R}_E(e^s) = b(e^s) - \sum_{\forall e^v \uparrow e^s} b(e^v)$$

Thus, the more widely applied constraints on mapping virtual nodes to substrate nodes is as follows

$$c(n^v) \leq c(n^s) - \mathcal{R}_N(n^s)$$

For the virtual link mapping procedure, there is a little difference from the above node mapping procedure. The fact is that one virtual link usually was mapped onto several substrate links end to end, which are called path. Thus, the remaining bandwidth of the path is decided by the smallest one.

$$\mathcal{R}_E(P) = \min_{\forall e^s \in P}(e^s)$$

Therefore, the requirements must be satisfied in virtual link mapping procedure should be

$$b(e^v) \leq \mathcal{R}_E(P), \forall P \in \{\textit{All the mapped substrate links}\}$$

3.3 Network Description

A virtual network is similar as the substrate network, which is consisted of virtual nodes and virtual links. So the most fittable notations are $G^v = (N^v, E^v)$, and G^v denotes the virtual network topology, N^v denotes the set of nodes in virtual network, E^v denotes the set of virtual links of virtual network. The requirements of computation of each virtual node is described by $c(n^v)$, where $n^v \in N^v$ represents the subset of virtual nodes. In the same way, the bandwidth of each virtual link is marked by $b(e^v)$, where $e^v \in E^v$, and denotes the subset of virtual links in virtual network.

3.4 VN Embedding Problem Description

When a virtual network request arrives, the substrate network should decide whether to accept the request or not. If the request accepted, a suitable mapping solution should be proposed by the substrate network. Generally, there are two phases in the mapping procedure, which are virtual node mapping and virtual link mapping. The node mapping phase usually is followed by the virtual link mapping phase.

Figure 1 describes the process of virtual network request being mapped to the substrate network with node constraints and link constraints. The two virtual network request finally mapped to the two separate part of the substrate network. This is merely an example of virtual network mapping, the more common case is several virtual network sharing the same part of the substrate network. The leftside of figure 1 depicts the virtual network request 1 and 2, which need to be mapped onto the substrate network on the right side. The numbers in

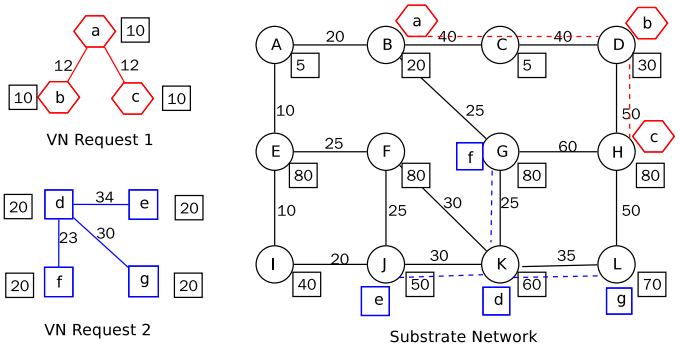

Fig. 1. Mapping virtual network request to substrate network.

black rectangle represent the computaton capacity, in more details, that in virtual network is the VN request's requirements to the physical nodes and that in substrate network is the computation capacity the physical nodes can offer. Similarly, the number on lines shows the bandwidth of links, one for requirements, and the other one for the provided.

Virtual network embedding problem is defined as

$$\mathcal{M} : G^v(N^v, L^v) \rightarrow G^s(N^s, L^s)$$

from G^v to a subset of G^s, where $N^v \subset N^s$ and $L^v \subset L^s$.

Node Mapping Virtual nodes from the same virtual network must be mapped to different substrate nodes. In the mapping $\mathcal{M}_\mathcal{N} : N^v \rightarrow N^s$, for all n^v, $m^v \in N^v$,

$$\mathcal{M}_\mathcal{N}(n^v) \in N^s$$

$$\mathcal{M}_\mathcal{N}(m^v) = \mathcal{M}_\mathcal{N}(n^v), if f m^v = n^v$$

subject to

$$c(n^v) \leq c(\mathcal{M}_\mathcal{N}(n^v)) - \sum_{i^v \in N^v} c(i^v)$$

$$dis(loc(n^v), loc(\mathcal{M}_\mathcal{N}(n^v)) \leq D$$

where *dis(a,b)* is the distance of node a and node b in substrate network.

In figure 1, the first VN request has the node map of { a $\rightarrow$ B, b $\rightarrow$ D, c $\rightarrow$ H}, and the second VN request has the node map of { d $\rightarrow$ K, e $\rightarrow$ J, f $\rightarrow$ G, g $\rightarrow$ L }. Note that in this case, there is no virtual nodes from different VN request sharing the same substrate node. But it may be appeared that virtual nodes from different VN request be mapped onto the same substrate node in the real virtual network environments.

Link Mapping The virtual links from VN request should be mapped to one or more substrate links with path splitting and migration. It is defined by a mapping $\mathcal{M}_E : \mathcal{E}^v \rightarrow \mathcal{P}^s$ from virtual links to a subset of substrate links such that for all $e^v = (m^v, n^v) \in L^v$,

$$\mathcal{M}_E(m^v, n^v) \subseteq \mathcal{P}^s(\mathcal{M}_N(m^v), \mathcal{M}_N(n^v))$$

subject to

$$b(e^v) \leq \mathcal{R}_E(\mathcal{P}), \forall \mathcal{P} \in \mathcal{M}_E(e^v)$$

3.5 Objectives

For the work in this paper, the major concentration is on improving the performance of the embedding procedure such as increasing the revenue of the virtual network and decreasing the cost of vitual network embedding which is similar to all the previous works in [4,8,9,11]. Besides, we also try our best to apply the network virtualization into wireless environments, more accurately, the Moile Ad Hoc Network environments, which is different from the works in [13–15].

Similar to previous efforts [4, 8, 9, 11],the revenue of the virtual network request as :

$$\mathcal{R}(G^v) = \sum_{e^v \in E^v} b(e^v) + \sum_{n^v \in N^v} c(n^v)$$

While revenue gives an insight into how much an InP will gain by accepting a VN request, it is not very useful without knowing the cost the InP will incur for embedding that request. We define the cost of mapping a virtual network request as the sum of total resources allocated to virtual network.

$$\mathcal{C}(G^v) = \sum_{e^v \in E^v} \sum_{e^s \in E^s} f_{e^s}^{e^v} + \sum_{n^v \in N^v} c(n^v)$$

where $f_{e^s}^{e^v}$ denotes the total bandwidth allocated on substrate link e^s for virtual link e^v.

4 Algorithms of VN Embedding

In graph thoery [16], each node in the undirected graph has a degree noting that how many links are associated with. With the notion of node degree in mind, we divide the VN embedding into two greedy stage as most previous efforts. For the first stage, the node mapping algorithm take the effect, in which each node of the virtual network will be sorted by its degree in descending order. The reason for such processing is that the larger of the node degree means that the node will be more important in the virtual network. Further more, the node with larger degree being mapped onto the substrate network, the consequence nodes will be mapped more easily. In addition, such dealing with node mapping will increase the probability of accepting the virtual network request. The calculations of node degrees is as follows.

Algorithm 1 Calculations of node degrees

Require: The set of virtual network request $G^v = (N^v, E^v)$
Ensure: The vector of virtual nodes with descending order on node id $vNodeRank$

```
for each e^v ∈ E^v do
    id← Get the first vertex id of e^v
    vNodeRank[id]++
    id← Get the second vertex id of e^v
    vNodeRank[id]++
end for
return vNodeRank;
```

The notion is similar with the idea in [11] on a certain extent, but the most obvious difference is that the virtual nodes with highest node ranking will be mapped onto the substrate nodes with highest node ranking in [11]. In our work, the virtual nodes with highest node ranking might not be mapped onto the substrate nodes with the same highest node ranking. We sort both the nodes in virtual network request and the substrate network by the note degree which is decided by the associated links in descending order. In order to increase the accepted ratio of VN request and improve the utilization of substrate nodes, we map the virtual nodes with highest degree and lowest cpu requirements to the substrate nodes who obtains the highest degree in substrate network environments. The virtual node mapping algorithm is as follows.

Algorithm 2 Virtual node mapping algorithm

Require: The set of virtual network request $G^v = (N^v, E^v)$
Ensure: The sequence of virtual nodes being mapped onto substrate nodes

```
Sort the virtual nodes N^v by node degree
Sort the substrate nodes N^s by node degree
for each n^v ∈ N^v do
    maxdegree=degree(n^s)*1.2
    mindegree=degree(n^s)*0.8
    for each n^s ∈ N^s do
        if degree(n^s) ∈ (mindegree, maxdegree) then
            potentialNodeSet ← n^s
        end if
    end for
    find n^s ∈ potentialNodeSet with least remain computing resource
    n^s ← n^v
end for
```

For the virtual link mapping phase, we use the greedy algorithm as follows.

Algorithm 3 Virtual link mapping algorithm

Require: The set of VN request $G^v = (N^v, E^v)$, with all the edges in Q
Ensure: The state that shows the mapping procedure succeeded or failed.

```
while Q ≠ ∅ do
    E^v=Q. dequeue();
    Remove those substrate links that cannot satisfy the bandwidth requirement of
    E^v. Use the shortest path algorithm to find a link mapping solution for E^v.
    if cannot find a path for E^v then
        return FAILED
    end if
end while
return SUCCESS
```

5 Performance Evaluations

In this section, we first describe the performance evaluations and then present our main evaluations result with analysis. Our evaluations mainly on the ratio of accepting VN request for substrate network, the revenue of the VN and the utilization of substrate nodes and links.

5.1 Simulation Environments

Our VN embedding simulation environment is based on the simulator called ViNE-Yard. It is a discrete event simulator with about five thousand lines of code implemented by C++ language. The simulator is freely available in the address [17]. According to the previous work [8], the substrate network topology in our experiments are randomly generated with 50 nodes using the GT-ITM tool [18] (25×25) grids. 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. It's assumed that VN requests arrive in a Poisson process with an average rate of 4 VNs per 100 time units, and each one has an exponentially distributed lifetime with an average of $= 1000$ time units. In each VN request, the number of virtual nodes is randomly determined by a uniform distribution between 2 and 10 following similar setups to previous works [4, 6]. The average VN connectivity is fixed at 50%. The CPU requirements of the virtual nodes are real numbers uniformly distributed between 0 to 20 and the bandwidth requirements of the virtual links are uniformly distributed between 0 to 50.

5.2 Result Analysis

There are four performance metrics which are described in the former section, the accepted ratio of VN request, the revenue of VN, and utilization of substrate nodes and substrate links. The main observations we want to summarize are as follows. Firstly, figure 2 shows the changing of VN request accepted ratio

over time. From the very beginning, due to the sufficient resources, both the computing resources and bandwidth in substrate network, the VN request is quickly accepted by the substrate network. While, with the passage of time, the available resources in substrate network decreases dramatically, which causes the drop down of the VN accepted ratio. After a period, the balance is established between the VN request incoming and VN cancelling. From the figures 3,4,5, it can be seen that the utilization of substrate nodes and substrate links has an obvious improving space.

6 Conclusions and Future Work

Besides the frameworks of virtual network systems, the major problems in network virtualization is the virtual network embedding problem. Improving the acceptance ratio of virtual network request is the most important aspect, not only because it can increase the revenue but also make full utilization of the resources in substrate networks. In this paper, we devised a virtual network mapping algorithm based on the similarity between the nodes in virtual networks and that in substrate networks.

The unstable link state in MANET environments is caused by the movements of nodes, we just define the link state categories without deep into the regular pattern of node movements in the paper. But the estimation against the link state

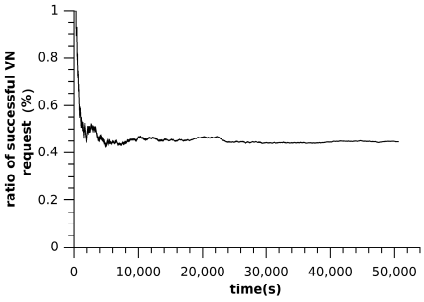

Fig. 2. VN request accepted ratio over time

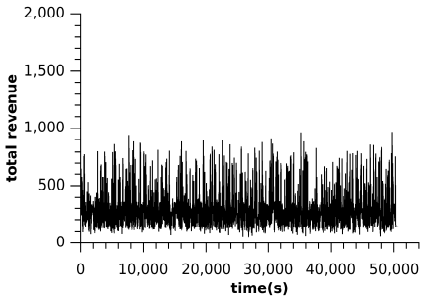

Fig. 3. VN Revenue over time

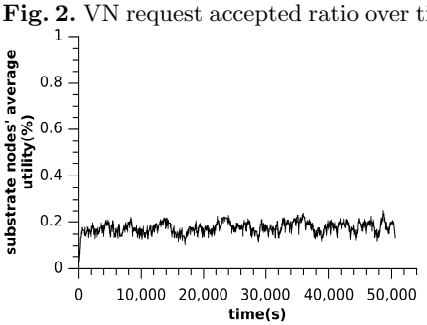

Fig. 4. Substrate nodes' average utility

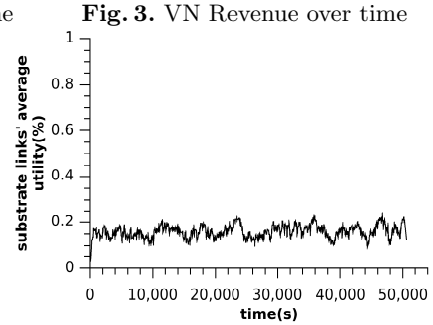

Fig. 5. Substrate links' average utility

is not accurate enough, we shall discover the distributions of node mobility. For example, let the node follow the random walk movement model or some other models. In other words, the link state should be estimated by finding the accurate assessment of node movement.

Current works all define the problem space by fixed CPU capacity and link bandwidth, but in the real world, they are varying all the time. So there is still spaces on improving the usage of the computing resources and bandwidth resources. What's more, the standard on selecting the appropriate nodes and links for virtual network be mapped to is various, current major works just define a fixed one on the selection standard, this is not widely applied in real environments. From the previous works, researchers propose the methods on path migration and splitting, which raise large amount of the ratio on virtual network request acceptance. While if the node migration and splitting is supported, then a virtual network request is rejected only if the total amount of bandwidth and the ability of computing required by virtual networks exceed from that of substrate networks.

Acknowledgments

We would like to thank the anonymous reviewers for their comments and suggestions. This work is supported by the project of the National Science Foundation of China under Grant No. 61073076, Ph.D. Programs Foundation of Ministry of Education of China under Grant No.20121102110018, the 2013 Open Funds of Science and Technology on Information Transmission and Dissemination in Communication Networks Laboratory, and the Postgraduate Innovation Practice Fund (YCSJ-02-02) in Beihang University.

References

[1] T. Anderson, L. Peterson, S. Shenker, and J. Turner, "Overcoming the internet impasse through virtualization," *Computer*, vol. 38, pp. 34 – 41, april 2005.

[2] J. Turner and D. Taylor, "Diversifying the internet," in *Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE*, vol. 2, pp. 6 pp. –760, dec. 2005.

[3] N. Chowdhury and R. Boutaba, "Network virtualization: state of the art and research challenges," *Communications Magazine, IEEE*, vol. 47, pp. 20 –26, july 2009.

[4] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: substrate support for path splitting and migration," *SIGCOMM Comput. Commun. Rev.*, vol. 38, pp. 17–29, Mar. 2008.

[5] I. Houidi, W. Louati, and D. Zeghlache, "A distributed and autonomic virtual network mapping framework," in *Autonomic and Autonomous Systems, 2008. ICAS 2008. Fourth International Conference on*, pp. 241 –247, march 2008.

[6] Y. Zhu and M. Ammar, "Algorithms for assigning substrate network resources to virtual network components," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pp. 1 –12, april 2006.

[7] G. Alkmim, D. Batista, and N. Saldanha da Fonseca, "Optimal mapping of virtual networks," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pp. 1 –6, dec. 2011.

[8] M. Chowdhury, M. R. Rahman, and R. Boutaba, "Vineyard: virtual network embedding algorithms with coordinated node and link mapping," *IEEE/ACM Trans. Netw.*, vol. 20, pp. 206–219, Feb. 2012.

[9] N. Chowdhury, M. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," in *INFOCOM 2009, IEEE*, pp. 783 –791, april 2009.

[10] A. Schrijver, "Theory of linear and integer programming," in *New York: Wiley*, 1986.

[11] X. Cheng, S. Su, Z. Zhang, H. Wang, F. Yang, Y. Luo, and J. Wang, "Virtual network embedding through topology-aware node ranking," *SIGCOMM Comput. Commun. Rev.*, vol. 41, pp. 38–47, Apr. 2011.

[12] I. Houidi, W. Louati, and D. Zeghlache, "A distributed virtual network mapping algorithm," in *Communications, 2008. ICC '08. IEEE International Conference on*, pp. 5634 –5640, may 2008.

[13] D. Yun and Y. Yi, "Virtual network embedding in wireless multihop networks," in *Proceedings of the 6th International Conference on Future Internet Technologies*, CFI '11, (New York, NY, USA), pp. 30–33, ACM, 2011.

[14] R. Kokku, R. Mahindra, H. Zhang, and S. Rangarajan, "Nvs: a virtualization substrate for wimax networks," in *Proceedings of the sixteenth annual international conference on Mobile computing and networking*, MobiCom '10, (New York, NY, USA), pp. 233–244, ACM, 2010.

[15] R. Kokku, R. Mahindra, H. Zhang, and S. Rangarajan, "Nvs: a substrate for virtualizing wireless resources in cellular networks," *IEEE/ACM Trans. Netw.*, vol. 20, pp. 1333–1346, Oct. 2012.

[16] "Graph thoery." `http://en.wikipedia.org/wiki/Graph_theory`.

[17] N. Chowdhury, "Vine-yard simulator." `http://www.mosharaf.com/ViNE-Yard.tar.gz`.

[18] E. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *INFOCOM '96. Fifteenth Annual Joint Conference of the IEEE Computer Societies. Networking the Next Generation. Proceedings IEEE*, vol. 2, pp. 594 – 602 vol.2, mar 1996.