Virtual Network Embedding For Evolving Networks

Zhiping Cai, Fang Liu, Nong Xiao, Qiang Liu, Zhiying Wang
School of Computer, National University of Defense Technology
410073 Changsha, Hunan, P.R.China
Email:zpcai@nudt.edu.cn,liufang@nudt.edu.cn,nongxiao@nudt.edu.cn,lqabc110@sina.com, zywang@nudt.edu.cn

***Abstract*—Network virtualization has been proposed as a powerful vehicle for running multiple customized networks on a shared infrastructure. Virtual network embedding is a critical step for network virtualization that deals with efficient mapping of virtual nodes and virtual links onto the substrate network resources. Previous work in virtual network embedding primarily focused on designing heuristic algorithms for static networks. Virtual network infrastructure should be reconfigured or redeployed in response to network growth. In this paper, we address the problem of optimally redeploying the existing virtual network infrastructure as the network evolves. This problem focus on minimizing the upgrading cost of virtual network, with satisfying node resource constraint and path delay constraint. It is shown that this problem is NP-hard. A heuristic algorithm is proposed and its effectiveness is validated by simulations evaluation.**

I. Introduction

Network virtualization is a networking environment that allows multiple service providers to dynamically compose multiple heterogeneous virtual networks, and to deploy customized end-to-end services as well as to manage them on those virtual networks for the end-users[1]. Network virtualization provides a powerful way to run multiple networks over a shared substrate. Network virtualization also plays an important role to support multiple architectures simultaneously as a long-term solution for the future Internet [2], [3].

A major challenge of network virtualization is the virtual network embedding problem that deals with efficient mapping of virtual nodes and virtual links onto the substrate network resources. Recent proposals for virtual network embedding problem focus on designing heuristic algorithms and customized algorithms for efficiently assigning virtual network to substrate resources[3], [4], [5], [7], [8], [9].

Previous work in virtual network embedding primarily focused on designing efficient mapping strategy for only considering fixed network topologies. In this paper, we address the problem of optimally upgrading the existing virtual network as the network evolves. The problem focus on minimizing the network operation cost with satisfying resource constraints.

When the network changes in response to network growth, node failures or node joining/leaving, the network topologies including nodes and links may change. Consequently, some virtual node may be migrated because there is not enough bandwidth or low delay on the new path to satisfy the resource requirement of virtual network. Some virtual node must be remapped as its substrate nodes have been deleted from the network. As a result, to be able to reconstruct the virtual network in new network, some existing virtual nodes may need to be migrated or reconfigured, or some virtual nodes may need to be remapped and deployed. The optimal problem of virtual network embedding for evolving networks is to minimize total upgrading cost, including migrating and remapping cost, while satisfying node resource constraints and the link delay constraints.

Each virtual network request has resource constraints, such as processing resources on the nodes and bandwidth resources on the links, that the embedding must satisfy. For example, to run a controlled experiment, a researcher may need 1 GHz CPU for each virtual node and ensure the virtual path link delays less than 500 msec. The combination of node and link constraints make the embedding problem computationally difficult to solve. The optimal problem formulation of virtual network embedding with resource constraints for evolving networks is presented and we provide the integer programming formulation. As the optimal problem is NP hard, we develop a heuristic algorithm and evaluate its performance using simulated network evolution scenarios.

The rest of the paper is organized as follows: related work is discussed in Section II. In Section III, we describe the problem formulation. In next section, we give a heuristic algorithm to solve the optimized network virtualization embedding problem for evolving networks. The effectiveness of the heuristic algorithm is validated by simulations evaluation in Section V. Finally, conclusions are presented in the last section.

II. RELATED WORK

There are a number of studies attempting to address the issue of efficiently assigning virtual network to substrate resources. In order to reduce the hardness of the virtual network assignment problem and to enable efficient heuristics, existing researches have been restricting the problem space in different dimensions [9]. These studies focus on specific virtual topologies [5], or only considered the offline version [5], [4], or ignored either node or link requirements [5] and [6], or assume infinite capacity of the substrate nodes and links [4],[6]. More recently, [3] and [9] have assumed that path splitting and node migration are supported by the substrate network.

All of the above approaches do not address the issue of evolving network. With the increasing pace of network growth, there is a need to evolve the existing virtual network infrastructure to satisfy the demands of the new network topology. To the best of our knowledge, this paper is the first

978-1-4244-5638-3/10/$26.00 ©2010 IEEE

attempt to address the optimizing problem of virtual network embedding for evolving network.

III. System Model and Problem Formulation

A. Substrate Network

We use two undirected graphx, $G_1 = (V_1, E_1, R_1^v, R_1^e)$ and $G_2 = (V_2, E_2, R_2^v, R_2^e)$, to represent the original substrate network and the evolved substrate network, respectively. We use subscript to refer to the type of network, and use superscript to refer to nodes or links. V_1 and V_2 are the sets of substrate nodes that can be mapped in G_1 and G_2, respectively. E_1 and E_2 are the set of links. Substrate nodes and links are associated with their attributes, i.e . the resources can be provided, denoted by node resource R_1^v and link resource R_1^e for the original substrate network, and node resource R_2^v and link resource R_2^e for the evolved network, respectively.

In this paper, we consider CPU capacity for node attributes, and link delay for link attributes. Fig. 1(b) shows a substrate network, where the numbers over the links represent link delay and the numbers in rectangles are the CPU resources available at the nodes.

The link delay constraint is considered in our model. Link delay represents the bandwidth constraint and capacity for each link. The edge-delay function $D(e)$ is designed which assigns a non-negative weight to each link in the network. The value $D(e)$ associated with edge is a measure (estimate) of total delay that packets experience on the link. It is assumed all pathes in substrate networks are loop-free paths. Let the set $E(Path(u, v))$ represent the set of edges which are in the shortest path from node u to node v , so we would have $Delay(Path(u, v)) = \sum_{e \in E(Path(u,v))} D(e)$.

B. Virtual Network Requests

We use an undirected graph, $H = (W, F, C_H^v, C_H^e)$ to represent the virtual network. A virtual network request typically has link and node constraints that are specified in terms of attributes of the substrate network. We use C_H^v and C_H^e to represent the set of node and link constraints, respectively. Fig. 1(a) depicts two virtual requests: the virtual network request 1 requires the path delay between the each node pair (a, b),(a, c) and (b,c) being under 20, and the CPU resource 10,20,25 at node a, b, and c, respectively. The virtual network request 2 requires the path delay between node d and node e should be under 20, and the CPU resource 5, 15 at node d and e, respectively.

C. Virtual Network Embedding

A virtual network embedding for virtual network requests is defined as a mapping *M* from H to a subset of G, such that the constraints in H are satisfied, i.e., Virtual Network Mapping:

$$M : (W, F, C_H^v, C_H^e) \longmapsto (V_2', P_2', R_2^v, R_2^e) \qquad (1)$$

where P_2' is the subset of substrate path,$V_2' \subset V_2$, and R_2^v and R_2^e are the node and link resources allocated in G_2 for the virtual network requests. Fig.1(b) shows the virtual network

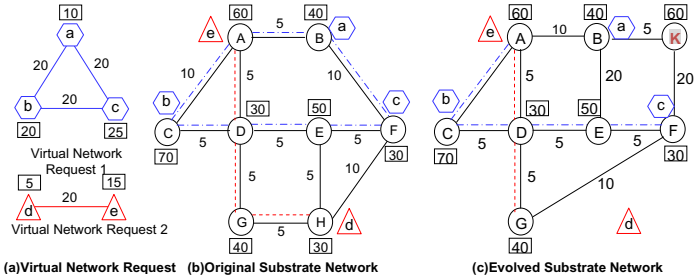

Fig. 1: Virtual Network Request, Original Substrate Network and Evolved Substrate Network

embedding solutions for the two requests. For example, the virtual node a, b and c in virtual network request 1 are mapped to the substrate nodes B, C and F, and the virtual links (a, b), (a, c) and (b, c) are mapped to the substrate paths $< BA, AC >$, $< BF >$ and $< CD, DE, EF >$ with the CPU and link constraints both satisfied. For virtual network request 2, the virtual node d and e are mapped to the substrate nodes H and A, and the virtual link (d, e) is mapped to the substrate path $< HG, GD, DA >$.

We use the a collection of indicator variables o_{ij} and x_{ij} to represent the mapping relationship of virtual network H with the original substrate network G_1 and the new network G_2, respectively. $o_{ij} = 1$ if and only if virtual network node $v_i \in W$ is mapping to substrate network node $v_j \in V_1$, otherwise $o_{ij} = 0$. Similarly, $x_{ij} = 1$ if and only if network node $v_i \in W$ is mapping to network node $v_j \in V_2$, otherwise $x_{ij} = 0$.

D. Cost Function

When the network changes from G_1 to G_2, the path between these virtual node may change. Consequently, some virtual node must be migrated because there is not enough bandwidth or low delay on the new path to satisfy the virtual network resource requirement. Some virtual nodes must be remapped as its substrate nodes have been deleted from the substrate network. In Fig. 1(C), node H has been deleted and node K has been added. The link (G, H),(E, H),(H, F) and (B, F) have been deleted and link (G, F),(B, E),(B, K) and (K, F) have been added. Consequently, the virtual node d must be remapped as the substrate node H has been deleted. The virtual node c must be migrated as the delay of path (A, F) is beyond delay constraint value 20.

As a result, to reconstruct the virtual network in new network, some existing virtual node may need to be migrated or reconfigured, or some virtual nodes may need to be remapped and redeployed. Clearly, the costs of migrating a virtual node is different from the cost of remapping a new virtual node. We use a parameter ρ to capture the ratio between these two. The total cost of reconstruct virtual network in response to network change is the sum of the cost for newly remapping virtual nodes and that for migrating existing virtual nodes. Since cost is an expense to a network operator, it is desirable to minimize it.

978-1-4244-5638-3/10/$26.00 ©2010 IEEE

E. Optimal Objectives

We define some binary variables to indicate whether the substrate nodes are located the virtual node. If one substrate node is located a virtual node, it is called "location node". The indicate variable $y_j = 1$ if substrate node v_j is a location node in the evolved substrate network, and 0 otherwise. $p_j = 1$ if substrate node v_j is a location node in the original substrate network, and 0 otherwise.

For all original location nodes in node set $V_1 \cap V_2$, there are two instances in the reconstruct process: 1)the location nodes are not changed which number is $\sum_{\forall v_i \in W, v_j \in (V_1 \cap V_2)} o_{ij} x_{ij}$; 2)the locating virtual nodes must be migrated. Since the total number of original location nodes in $V_1 \cap V_2$ is $\sum_{\forall v_i \in W, v_j \in (V_1 \cap V_2)} p_j$, the number of virtual nodes being migrated is $\sum_{\forall v_i \in W, v_j \in (V_1 \cap V_2)} p_j - \sum_{\forall v_i \in W, v_j \in (V_1 \cap V_2)} o_{ij} x_{ij}$. All the virtual nodes located in the location nodes that were deleted from V_1 must be remapped and its number is $\sum_{\forall v_j \in (V_1 - V_2)} p_j$. So we can formally define the cost function as:

$$C(\rho, G_1, G_2) = \sum_{\forall v_i \in W, v_j \in (V_1 \cap V_2)} (p_j - o_{ij} x_{ij}) + \rho \sum_{\forall v_j \in (V_1 - V_2)} p_j. \tag{2}$$

F. Integer Programming Formulation

The optimal problem of virtual network embedding for evolving network can be stated as follows: Given original network G_1 and new network G_2, existing virtual network mapping relationship $O = \{o_{ij} | v_i \in W \& v_j \in V_1\}$ in G_1, determine a virtual network mapping scheme $X = \{x_{ij} | v_i \in W \& v_j \in V_2\}$ in G_2 such that:

(1)the cost of changing from O to X is minimum; (2)all virtual nodes in H are mapped in substrate network G_2; (4)the node resource request of all located virtual nodes should be satisfied; (4)the path delay on each virtual link should be less than the maximum delay bound δ;

The optimal problem of minimizing the virtual network reconstruct cost for evolving network subject to delay constraints can naturally expressed as an optimization problem. The objective is:

$$MinimizeC(\rho, G_1, G_2) \tag{3}$$

and the subjects are below:

$$\sum_{j=1}^{|V_2|} x_{ij} = 1(\forall i | v_i \in W, v_j \in V_2,) \tag{4}$$

$$x_{ij} \leq y_j (\forall v_i \in W, v_j \in V_2) \tag{5}$$

$$x_{ij} C_H^v(v_i) \leq x_{ij} R_2^v(v_j) (\forall v_i \in W, v_j \in V_2) \tag{6}$$

$$Delay(Path(v_j, v_k)) y_j y_k \leq \delta (\forall v_j, v_k \in V_2) \tag{7}$$

$$x_{ij} \in \{0, 1\} (\forall v_i \in W, v_j \in V_2) \tag{8}$$

$$y_j \in \{0, 1\} (\forall v_j \in V_2) \tag{9}$$

The constraint (4) and (5) ensure that each virtual node is mapped to one substrate node in G_2 exactly. The constraint (6) ensures that located substrate node can satisfy the virtual node resource request. The constraint (7) ensures that path delay of each pair virtual nodes not exceeds the delay constraint.

G. Computational Complexity Issues

The optimal problem of virtual network embedding for evolving networks can be proved NP-hard via a reduction from the Maximum Independent Set problem to this optimal problem. The optimal problem of virtual network embedding for evolving networks is also a special case of the virtual network embedding problem. The virtual network embedding problem with satisfying multiple objectives and constraints can be formulated as an NP-hard problem. Thus we do not hope to get an exact polynomial time algorithm for optimal problem of virtual network embedding for evolving networks.

IV. Heuristic Algorithms

Section III shows that the problem of obtaining optimal virtual network adjustment strategy under delay constraint in evolving networks is NP-hard. We therefore examine polynomial time heuristics in this section. The objective of this heuristic is to minimize the cost function, which includes both migrating virtual node cost and remapping virtual node cost. The value ρ is the ratio of the relative cost of migrating a new virtual node vs. remapping a virtual node and thus provides the flexibility to tune the cost function for any given network scenario. This similar method is also used in [10] to solve the optimize problem of distributed Network monitoring. The size and complexity of each network is different, resulting in different cost of migrating and remapping virtual nodes. Therefore, the value of ρ can be different for each network operator. ρ may also change over time, as hardware price fluctuates, migration technology and remapping schemes evolves. We designed our heuristic in such a way that it can work well with a wide range of ρ. In some scenariosnode migration may not consume much time. [11] shows that they are able to migrate virtual machine in a few seconds, therefore we can assume the migration cost is 0 in this scenario, i.e the total cost is determined only by the remapping cost.

The heuristic algorithm presented in this section is to minimize the cost function C, while ensuring that all virtual node are assigned to a substrate node and the resource constraint and delay constraint are both satisfied. Our heuristic algorithm consists of three steps. In the first step, our proposed algorithm calculate out the constraint cover set of each substrate node, i.e. the set of substrate nodes satisfying the delay constraint for each substrate node. In the second step we locate the virtual node which must be migrated by using greedy strategy. The heuristic algorithm greedily pick the substrate node having enough node resource to satisfy the virtual node constraint, while the node number of its constraint cover set is maximum in the candidated node set. In the three step we remap the unmapped virtual nodes to substrate nodes using greedy strategy. Similarly to the second step, the greedy algorithm picks the substrate node having enough node resource to

978-1-4244-5638-3/10/$26.00 ©2010 IEEE

satisfy the virtual node constraint, while the node number of its constraint cover set is maximum in the candidate node set.

A. Compute Constraint Cover Set

The first step of our heuristic algorithm computes the constraint cover set for each substrate nodes. The formal description of constraint cover set of relative substrate node is below: Given the delay constraint δ, the constraint cover set of substrate node v, denoted as $C(v) \subseteq V$, with satisfying that: for any node $u \in C(v)$, there is a path between u and v with $Delay(Path(u,v)) \leq \delta$. For any node in substrate network, we can use Breadth-First method to search maximum constraint cover set of each substrate node. The detail of computing constraint cover set algorithm are shown in Algorithm 1.

Algorithm 1 Computing Constraint Cover Set Algorithm

for $i = 1$ to $|V_2|$ **do**
  for $j = i + 1$ to $|V_2|$ **do**
    if $Delay(Path(v_i, v_j)) \leq \delta$ **then**
      $\{C(v_i) = C(v_i) \cup \{v_j\}, C(v_j) = C(v_j) \cup \{v_i\}\}$
return $C(v)$;

Algorithm 2 Migrating Algorithm

Find virtual nodes which request cann't be satisfied or conflict with the delay constraint, and add those nodes into M;
$U = \{v_j | (v_j \in V_2) \& (y_i \neq 1)\}$; //U is the set of candidate substrate nodes
Compute constraint cover set of each candidate substrate node with Algorithm 1;
Sort the candidate substrate node with decrease by the node number $|C(v_j)|$ of its constraint cover set;
while $M \neq 0$ **do**
  pick the substrate node v_j in U with the maximum node number $|C(v_j)|$;
  identify whether the resource of node v_j satisfy the request of virtual node v_i, If violate the node resource constraint, i.e. $R_2^v(v_j) < C_H^v(v_i)$, skip node v_j and goto step 6;
  identify whether the path delay between v_j and existent virtual nodes exceeds the delay constraint value. If violate the delay constraint, skip node v_j and goto 6;
  migrate the virtual node v_i to the substrate node v_j;
  $M = M - \{v_i\}$;
  $U = U - \{v_j\}, x_{ij} = 1$;
return X;

B. Migrate virtual nodes

The virtual nodes must be migrated if they conflict with the delay constraint or adjacent virtual links/nodes have been deleted. Our migration algorithm identifies whether the delay of each existent virtual path beyond the delay constraint. It would happen as link or path has been changed. Nodes which must be migrated are added into Migrating Node Set M. Our algorithm firstly sorts the candidated substrate node with decrease by the node number of its constraint cover set. For each virtual node in M, our algorithm greedily picks the candidate substrate node with the maximum number of nodes in its constraint cover set, while satisfying the delay constraint. If the picked candidate substrate node could not satisfy the node resource constraint, the algorithm will pick another one from the candidate substrate node. The detail of migration algorithm are shown in Algorithm 2.

C. Remapping Virtual Nodes

The virtual nodes must be remapped if their located nodes are deleted as network evolves. Nodes those must be remapping are added into Remapping Node Set R. For each virtual node in R, our algorithm greedily pick the location node with the maximum number of nodes in its constraint cover set, while satisfying the delay constraint. If the picked candidate substrate node can not satisfy the node resource constraint, the algorithm will pick the next one from the candidate substrate node set. The detail of node remapping algorithm are shown in Algorithm 3.

Algorithm 3 Remapping Algorithm

Sort the candidate substrate node with decrease by the node number $|C(v_j)|$ of its constraint cover set;
while $R \neq 0$ **do**
  pick the substrate node v_j in U with the maximum node number $|C(v_j)|$;
  identify whether the resource of node v_j satisfy the request of virtual node v_i, If violate the node resource constraint, i.e. $R_2^v(v_j) < C_H^v(v_i)$, skip node v_j and goto step 3;
  identify whether the path delay between v_j and existent virtual nodes exceeds the delay constraint value. If violate the delay constraint, skip node v_j and goto 3;
  map the virtual node v_i to the substrate node v_j;
  $R = R - \{v_i\}$
  $U = U - \{v_j\}, x_{ij} = 1$
return X;

V. Simulations

In this section, we evaluate the performance of the proposed heuristic algorithm on several different topologies and parameter settings.

A. Simulation Setup

We generate substrate network topologies by using the GT-ITM tool[12]. Different network topologies are generated by varying three parameters,n,α and β. Simulation results presented here are averaged over 5 different topologies.

The substrate network is configured to have 100 nodes and around 500 links, a scale that corresponds to a medium-sized ISP. The CPU resources of the substrate nodes are real number uniformly distributed between 30 and 100. The link delay follow a uniform distribution from 10 to 100 units. The evolution of the network is captured by stepwise randomly deleting 25 nodes and adding 50 nodes.

978-1-4244-5638-3/10/$26.00 ©2010 IEEE

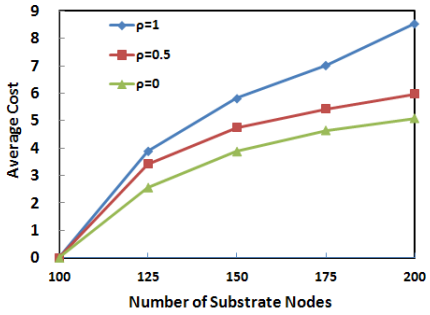

Fig. 2: Average cost as a function of the number of substrate nodes (the number of virtual nodes is fixed 10)

The virtual network topologies are random and the number of virtual nodes for one virtual network request is randomly determined by a uniform distribution between 2 and 10, following a similar setup of previous work[3], [4], [9].

B. Evaluation of the Adaptive Algorithm for Network Growth

Fig. 2 shows the average cost for upgrading the virtual network as a function of the stepwise random deleting 25 nodes and simultaneously adding 50 nodes. The three curves represent the instances with ρ equal to 1,0.5 and 0, respectively, while the number of virtual nodes is fixed 10. It is shown the upgrading cost is slowly increased. It is due to the fact that the network size is larger, the influence is smaller for virtual network infrastructure as the substrate network evolves when the virtual network requirement is not changed.

C. Performance Comparison of Algorithms

We compare the performance of our greedy algorithms with the optimal solution and the random algorithms. The optimal results are obtained by using integer programming solver. The random algorithms are using random picking strategy for migrating and remapping virtual nodes.

Table I presents one set of simulation results. we have obtained similar results for other parameter settings. Our results indicate that using our greedy algorithm is better than the random algorithm and the cost of our algorithm is close to the optimal results. It shows that our greedy algorithm is reasonably efficient.

TABLE I: Comparisons of Algorithms

$N_{substrate}$	$N_{virtual}$	$Cost_{optimal}$	$Cost_{greedy}$	$Cost_{random}$
150	20	9.3	9.9	13.4
150	30	16.1	17.1	22.5
200	20	13.6	14.2	18.9
200	30	21.3	22.6	26.0

D. Discussion

In some scenarios, multiple virtual nodes can be mapped to the same substrate node. It is possible that the substrate node and its adjacent links can satisfy the resource requests of multiple virtual nodes and virtual links. Our algorithm can account for this scenarios with minor modification: delete the step (10) from Algorithm 2 and delete the step (7) from Algorithm 3. It means that the substrate nodes can be hold in the candidate substrate node set even if it have been located virtual node. So the substrate nodes can be mapped to locate multiple virtual nodes if it can satisfy multiple virtual node resource requests.

In some networks it is possible that there are some fixed mapping between a virtual node v_i and a substrate node v_j as the business or zone demand. It is easily solved by setting $x_{ij} = 1$. Restrictions on mapping between v_i and v_j could be taken care easily as well by setting $x_{ij} = 0$. These restriction can reflect the fact that it is not desirable or impossible for mapping virtual nodes to a special substrate node. These special case can be deal with minor modification.

VI. Conclusion

In this paper, we have addressed the optimal problem of virtual network embedding for evolving networks. This problem focus to capture the difference between the two kinds of costs: migrating and remapping virtual node. We formulated the optimal problem as an integer programming problem.It is shown that the problem is NP-hard. The effectiveness of the proposed algorithm is validated by simulations evaluation.

Further research would be conducted to design adaptive heuristic algorithms for multiple objectives and satisfying multiple constraints. It would allow our approach to be applied in practice easily.

ACKNOWLEDGMENT

This work is supported by the National Basic Research Program of China (973) under Grant No. 2007CB310900, and partially supported by by NSFC under Grant No.60603062, No. 60903040. and NSFC-Hunan under Grant No.06JJ3035.

References

[1] N.M.Chowdhury, R.Boutaba, "Network Virtualization:The Past,The Present,and The Future", IEEE Communications Magazine,July,2009.
[2] J.Lischka, H.Karl, "A Virtual Network Mapping Algorithm based on Subgraph Isomorphism Detection", Proc. 1st ACM workshop on Virtualized Infrastructure Systems and Architectures (VISA 2009), August, 2009.
[3] M.Yu, Y.Yi, J.Rexford, M.Chiang, "Rethinking Virtual Network Embedding: Substrate Support for Path Splitting and Migration", ACM SIGCOMM Computer Communication Review, 2008,38(2), pp.19-29.
[4] Y. Zhu and M. Ammar,. "Algorithms for assigning substrate network resources to virtual network components. in Proc. IEEE INFOCOM, 2006.
[5] J. Lu and J. Turner,. "Efficient mapping of virtual networks onto a shared substrate. Washington University, Technical Report WUCSE-2006-35, 2006.
[6] J. Fan and M. Ammar, "Dynamic topology configuration in service overlay networks - a study of reconfiguration policies, in Proceedings of IEEE INFOCOM, 2006.
[7] I. Houidi, W. Louati, D. Zeghiache. "A Distributed Virtual Network Mapping Algorithm". Proc. IEEE ICC 2008, March 2008.
[8] Ines Houidi, Wajdi Louati, Djamal Zeghlache and Stephan Baucke, "Virtual Resource Description and Clustering for Virtual Network Discovery". Proc. ICC Workshop on the Network of the Future 2009, June, 2009.
[9] N. M. Mosharaf Kabir Chowdhury, Muntasir Raihan Rahman, Raouf Boutaba, "Virtual Network Embedding with Coordinated Node and Link Mapping", Proc. IEEE INFOCOM 2009, April 2009.
[10] Marina Thottan, Erran L. Li, Bin Yao, Vahab S. Mirrokni, Sanjoy Paul, "Distributed Network Monitoring for Evolving IP Networks", Proc. ICDCS 2004, March 2004.
[11] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Black-box and gray-box strate- gies for virtual machine migration", Proc. Networked Systems Design and Implementation, (Cambridge, MA), April 2007.
[12] E. W. Zegura, K. Calvert, and S. Bhattacharjee. "How to Model an Internetwork". In Proc. IEEE Infocom, volume 2, pages 594 602, 1996.

978-1-4244-5638-3/10/$26.00 ©2010 IEEE