Heuristic Algorithm for Virtual Network Mapping Problem

Huynh Thi Thanh Binh(✉), Bach Hoang Vinh, Nguyen Hong Nhat, and Le Hoang Linh

School of Information and Communication Technology, Hanoi University of Science and Technology, Hanoi, Vietnam
binh.huynhthithanh@hust.edu.vn

Abstract. Nowadays, resource allocation for virtual networks (VNs) is brought as an imperative problem. For the characteristics of virtual networks, multiple virtual networks with different topo can co-exist on a shared infrastructure. A difficult point of problem is how to use the resource effectively and to satisfy the requirement of virtual network request. This problem is NP-hard. In this paper, we introduce a heuristic algorithm to solve this problem. To simulate a virtual network mapping problem in real world, we using two input data: an infrastructure network which is modeled by a connected graph and a set of virtual network request graphs with each graph contains their constraints, time and duration. The main purposes are to maximize the revenue and to minimize the cost when allocate the virtual networks to a substrate network. The experimental results are reported to show the efficiency of propose algorithm comparing to the Enhanced Greedy Node Mapping (EGNM) algorithm.

Keywords: Substrate network · Virtual network · Resource allocation · Heuristic algorithm

1 Introduction

In recent years, the network resources are being depleted. IPv4 has reached the limit, while people are still indifferent to IPv6. The physical resources are increasing improve, but have not been use effectively, leading to waste of resources. So, the necessary matter now is to use it in a suitable and efficient way. A given solution is Network virtualization. That means we can create multiple Virtual networks (VNs) which has different topologies, protocols and services. But all of them can run on and share resource of the same infrastructure, network virtualization promises better flexibility, security, manageability and decreased power consumption for the Internet [1]. This paper considers two processes of solving. The first is construction problem by giving the inputs, outputs and objective of the work. Second is our proposed algorithm that we use to improve the solutions.

However, the difficulty of this problem is how to optimal the allocation from virtual network to physical network and take full advantage of physical resource. Hence, several algorithms have been proposed to seek for a near optimal solution while reducing the complexity of the problem [2].i.e. in [3], Son H.Ngo et al. have introduce an improved heuristics for online node and link mapping which is improvement of

Yu'algorithm [4] by adding three more steps: sorting virtual nodes, using adaptive function and pre-checking invalid substrate links. With the best of our knowledge, almost mappings nowadays just start handling when the new requests come. Thus, we introduce a new way to approach the problem, which will use a new algorithm to pre-compute to speed up the processing. In this work, we give some criteria to evaluate and the objectives of problem, as well as an algorithm to generate the requirement of VNs. In that algorithm, we propose some improvements: sorting nodes by their degree and their available resource and compute before the time requests come. This resulted in two goals: maximize the revenue (B) and minimize the Cost (C). To simplified the goal, we decided to use the ratio of revenue to cost (B/C) as the only objective function and try to maximize this ratio. Experiment on 50-node topology of the substrate network and a set of virtual network requests, expected positive results and better than EGNM algorithms and Yu's Baseline virtual network embedding algorithm.

The rest of this paper is organized as follows. It starts with the introduction of problem formulation and related works. We describe about virtual network mapping problem at Section 2 and review some existing algorithms, on Section 3. Our new proposed algorithm is showed in section 4. Section 5 gives the experiments, computational and comparative results. The paper concludes with discussions and future works in section 6.

2 Problem Formulation

In this section, we give some overviews about virtual network embedding problem as well as an algorithm to solve it. We will define general substrate network, virtual network and set the main point to this problem which derived mainly from Son H.Ngo [2, 3] and Yu [4].

Input: A substrate network and a set of virtual network request:

Substrate Network

We define the substrate network as a graph S = (N, L); where N is the set of substrate nodes and L is the set of substrate links of the network. Each node n ϵ N has an associated free CPU resource - cs and each link l ϵ L has an available bandwidth capacity - bs. Denoted by N = {n(cs)} and L = {l(bs)}.

From that, we can define the available resource of each substrate node by:

$$AR(n) = cs \times \sum_{l \in L(n)} bs \tag{1}$$

Where:- L(n) is the set of neighbor links of node n.

- cs is the available resource of node n.
- bs is bandwidth capacity of link l.

Virtual Network

The virtual network requests are modeled as r = (V, E, t, d) ϵ R, which is the set of requests. Where V, E are the set of vertices and edges of request, t is time when the request comes, and d is the duration of the request in turn.

Each vertex v ϵ V has a CPU capacity requirement - cr and each edge e ϵ E has a bandwidth capacity requirement - br. Denoted by V = {v(cr)} and E = {e(br)}.

Each virtual node of virtual network has its required resource define as:

$$RR(v) = cr \times \sum_{e \in E(v)} br(e) \tag{2}$$

Where:- E(v) is the set of neighbor edges of vertex n.

- cr(v) is the required resource of vertex n.
- br(e) is required bandwidth of edge e.

Constrains

- Satisfying the most of virtual network requests not using the most resource.
- Each VN request is served with highest benefit.

Output

A set of graphs s' = {s_1, s_2…s_n} where graph si is a result of mapping from virtual network r to physical network S.

Objectives

In this work, we are giving two main objectives: accept as many requests as possible (the acceptance ratio) and serve the VNs request with highest benefit brought while using the resource of physical network in optimal way. To do that, two quantities are given, that is revenue (B) and cost (C). And the objectives of this problem are maximizing revenue and minimizing the cost. However, it is difficult to satisfy both of them. So we decide to use the ratio of B to C to evaluate the final result.

The revenue here is determined by the total bandwidth of virtual link acceptance. The cost is evaluated by the number of substrate links of a path and the bandwidth was used in that path. In here, CPU resource is not the determining factor, and does not affect to the result of problem. Therefore, we decided to use only bandwidth to calculate the cost value.

- Maximize the Revenue calculate on set of request R:

$$B_{sum}(R) = \sum_{r_j \in R_A} B(r_j) = \sum_{r_j \in R_A} \sum_{e \in E(r_j)} br_e \tag{3}$$

Where: R_A is the set of requests that has accepted.

E(r_j) is the set of request in R_A.

- Minimize the Cost calculate on output s':

$$C_{sum}(s') = \sum_{s_j \in s'} C(s_j) = \sum_{s_j \in s'} \sum_{p \in P(s_j)} length(p) \times bw_p \tag{4}$$

Where length(p) equal the number of link on that path.

Simplified, the goal is maximize this equation:

$$O = \frac{B_{sum}}{C_{sum}(s')} = \frac{\sum_{r_j \in R_A} \sum_{e \in E(r_j)} br_e}{\sum_{s_j \in s'} \sum_{p \in P(s_j)} length(p) \times bw_p} \tag{5}$$

3 Related Works

The problem can be divided into two main stages: node mapping and link mapping [1, 3, 4].

In [5], Adil et al. has given three main constraints associated with Virtual Network Embedding (VNE) problem. That is Node constraint, Link constrains and Admission control. Each kind of these maintain many types of constraint that must be satisfied. E.g. for node constraints, there are capacity and location. For link constraint, there are bandwidth and link propagation delay. And admission control is an important constraint that need to be implemented for two reasons: It ensure that demands of newly arrived VNs can be fulfilled by the substrate; resource allocation made to already mapped VNs is not violated.

In [4], Yu et al. use two algorithms for two stages which are Greedy Node Mapping (GNM) and Link Mapping with k-shortest path Algorithm.

GNM process all request arriving within a time-window as well as in the request queue, in decreasing order base on their revenue. For each request, they map its virtual nodes to substrate nodes which has maximum available resource. The pros of this method is to minimize the use of substrate resources at bottleneck nodes, which helps in satisfying the requirements of future VN requests which demand fewer resources [5].

And for link mapping, the selected nodes are connected following k-shortest paths algorithm to form a completed virtual network. A found path is accepted if it has enough bandwidth.

In [2], Son H.Ngo was improved the GNM algorithm of Yu, also the link mapping method. Using Enhance Greedy Node Mapping (EGNM), he lower the cost of problem for large scale application. They not only sort the requests base on their revenue but also sort the virtual nodes according to their require resource before mapping them to substrate node. By this way, virtual nodes with more required resource will be mapped to substrate nodes which have more available resource. They also re-calculate the available resource (AR) by adding a threshold T that is define as the ratio of CPU Load to Link Load on substrate network. Since then, the AR value of a node is depend on T.

At second stage, the Link mapping with Pre-checking using Dijkstra's algorithm was used. To satisfy the bandwidth constraint, they propose a pre-checking scheme that verifies the status of resource usage in the substrate network before the link mapping. The substrate links that do not have more or equal available bandwidth than requests will be removed from the graph. Then, the Dijkstra's algorithm will only

search on the remaining links. This process helps to improve the acceptance ratio by assuring that once we find out a path, it will satisfy the bandwidth constraints [3].
In [6], Lischka et al. has combined two stages into only one stage. Then, with each incoming request, they used a backtrack algorithm to find a subgraph on substrate network that has the same form with virtual network. But a link on virtual network can be mapped with ε links on physical network. The disadvantage of this algorithm is that it waste more time to perform backtrack. Therefore it is not suited with the requests which have deadline to map.

Finally, the experiment's results in [3] show that, testing on the substrate node's size of 30 nodes and 50 nodes, the EGNM with pre-checked given the acceptance ratio and cumulative revenue better than GNM with k-shortest paths. But its revenue/cost ratio (B/C) is not good enough. Because the link mapping with Pre-checked has increased the cost by increasing the length of paths that satisfied the bandwidth constraints. Hence, decrease the B/C ratio.

According to this disadvantage, we propose a new method of node mapping and hoping it to simultaneously increase the revenue and decrease the cost while giving good acceptance ratio.

4 Proposed Algorithm

In this paper, we introduce a new way to approach the problem. Almost mappings nowadays just start handling when the new requests come. Thus, we propose a new algorithm to compute before meet a request to speed up the processing that is Path-Checking Adjacency Node Mapping (PCANM).

Some difficulties are the differences of the virtual networks' requirement and model. So, mapping them to the substrate network is also totally different. To simplified, we group the substrate node to the set with the same degree and add it into substrate list. This will help to predict the ability to provide resource of each node. Then, when the requests come, algorithm just has to search in this list instead of search in whole substrate network.

The arrangement, firstly, based on the degree of node, the nodes with the same degree will be sorted by their available resource. The virtual node, in another way, is only sorted by their required resource. The list is built before the arrival time of requests, auto update when a new request comes or another leaves.

Here are some briefly describe about PCANM. The ideal of this algorithm is simulating part of link mapping process while performing node mapping. Firstly, when the new requests come, we sort the virtual nodes in the order of descending of their required resource, and add it into list of virtual nodes. Then, take the one with largest required resource. For each virtual node was taken, we group the substrate node by their degree then compare the virtual node's degree with degrees of substrate nodes and the algorithm start processing from the group that has nearest greater than or equal with the virtual node's degree. In each group, the substrate nodes are sorted by their available resource in descending order. The final step before mapping is check whether the substrate node is satisfied the requirement of the virtual node or not and whether the substrate node (from the first one to the last one) is linked with other mapped substrate nodes or not and does any neighbors of this substrate node can be mapped with any neighbors of the virtual node or not.

Figure 1 describes the algorithm in general way. With two inputs: the virtual network r and substrate network S. At step 1, the algorithm finds and takes the virtual node which has highest required resource in r (the one has weight of 20). Then, we classify the substrate nodes into groups depends on their degree (step 2). At step 3, from the group which has the same degree with or nearest higher than the virtual node's degree, take the node with highest available resource (the one has weight of 35) to see if it can be mapped by the virtual node or not. The node that satisfied is the node has CPU resource greater than virtual node's CPU resource and has total bandwidth greater than virtual node's total bandwidth. If not, move to next substrate node in the group.

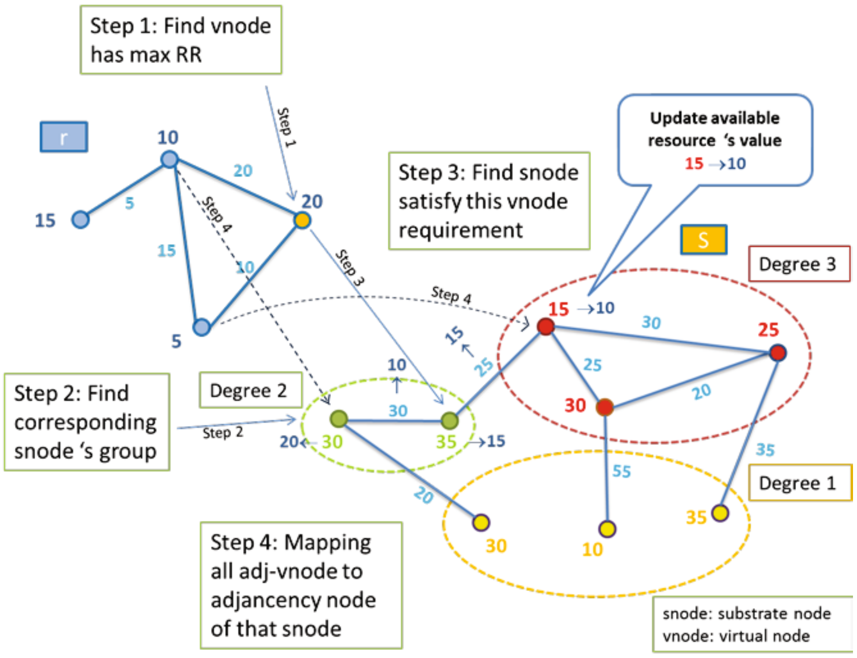

Step 1: Find vnode has max RR
r
Step 1
Update available resource 's value
15 → 10
Step 3: Find snode satisfy this vnode requirement
S
Degree 3
Step 4
Step 3
Step 4
Step 2: Find corresponding snode 's group
Step 2
Degree 2
Degree 1
Step 4: Mapping all adj-vnode to adjancency node of that snode
snode: substrate node
vnode: virtual node

Fig. 1. Visualizing algorithm

Step 3.1 is a significant improvement in this algorithm, with the virtual node and substrate node that is recently mapped, we check their neighbor nodes to see if which node can be mapped to which. For next virtual nodes, we add one more step before map it to the substrate node, which is called connection testing step. In this step, the substrate node about to be mapped will be checked if it has the link to other mapped node. If not, the process then stops and continues with another node.

Following is PCANM's pseudo-code. In this pseudo-code, vNode stands for virtual node and sNode stands for substrate node.

```
Algorithm: PCANM(S = (N,L), r = (V,E,t,d))

Input:     Graph of substrate network S=(N,L); and virtual
           network request r=(V,E,t,d)
Output:    Set of results of mapping s'=(s1',s2',…sn')
begin
  1.Sort vNode according to their Required Resource
  2.  for each vNode do
  3.    if vNode was not mapped yet
  4.    failed = true
  5.    Sort substrate node according to their degree
          and Available Resource
  6.      for each sNode do
  7.        if can map vNode to sNode
  8.        Mark vNode is now being mapped to sNode
  9.        if sNode have links to other mapped sNode
 10.          failed = false
 11.            Map neighbor nodes of vNode to neighbor nodes
                of sNode.
 12              break
 13.          end if
 14.        end if
 15.      end for
 16.      if failed
 17.        NodeMappingFailed()
 18.      end if
 19.    end if
 20. end for
end
```

5 Experimental Results

5.1 Problem Instances

The Datasets

Substrate network:

Size of substrate network is the number of its nodes. We create three different topologies with the large number of node to experiment, which is 50-node topology, 100-node topology, and 200-node topology. Each pair of substrate nodes is connected with probability 0.2.

CPU resources of nodes follow distribution from 1 to 50.

Bandwidth resources of nodes follow distribution from 10 to 60.

Virtual network:
Size of VN is between 3 and 6 nodes. Each pair of virtual nodes is connected with probability 0.2

CPU and Bandwidth resources of nodes follow distribution from 1 to 25.

The appearance ratio of virtual network request is 40%.

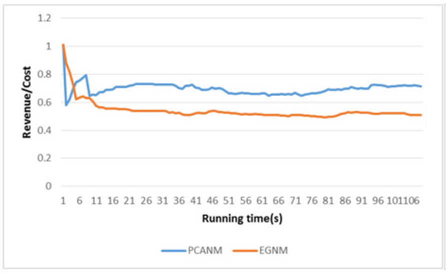
5.2 Experiment Setup

We implemented two algorithms EGNM and PCANM in Java using Eclipse IDE. Each algorithm is simulated twenty times with three different topologies of substrate network: 50-node, 100-node and 200-node topology. Then we take the average of twenty times as the final result to compare. The network is generated randomly by a program written in Java.

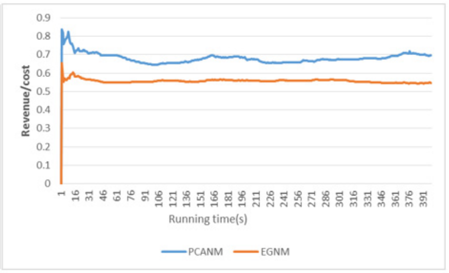
5.3 Computational Results

We evaluate the performance of these algorithms based on 3 criteria: revenue/cost ratio, acceptance ratio, and revenue. The compared data is obtained during the implementation process. The formulas to calculate revenue value and revenue to cost ratio was given in (3) (4) (5). Acceptance ratio is defined as the proportion of arriving virtual network requests that are accepted.

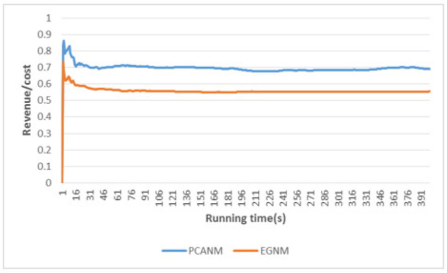
Fig 2 and 3 show that the approximate results between two algorithms, but the B/C ratio of proposed algorithm is much better than the old algorithm.



(a) $|N_S| = 50$

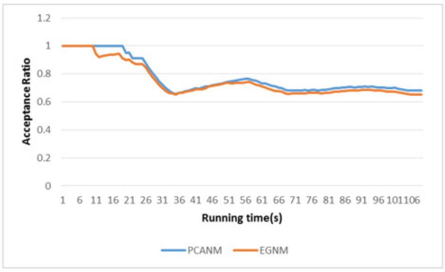


(b) $|N_S| = 100$

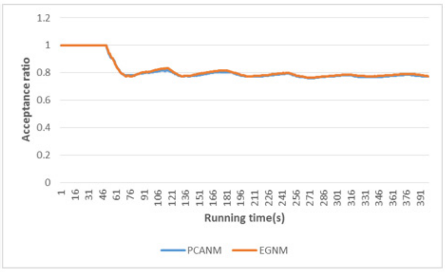


(c) $|N_S| = 200$

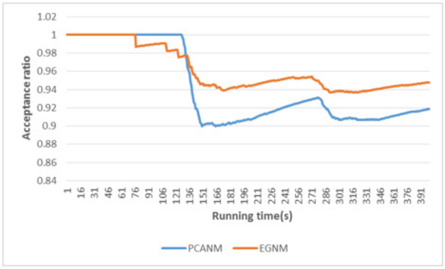
Fig. 2. Revenue/Cost (B/C) ratio compare between PCANM and EGNM based on the average data of twenty times running program



(a) $|N_S| = 50$

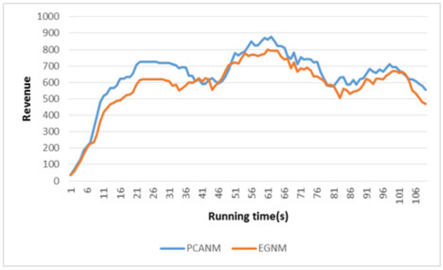


(b) $|N_S| = 100$

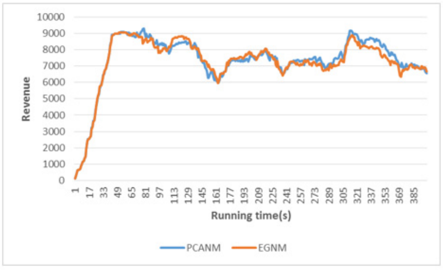


(c) $|N_S| = 200$

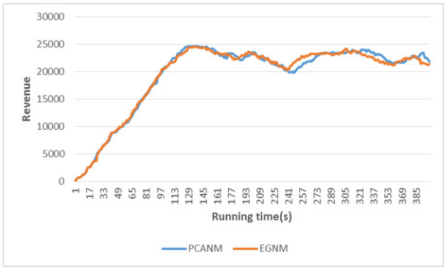
Fig. 3. Acceptance ratio compare between PCANM and EGNM based on the average data of twenty times running program



(a) $|N_S| = 50$



(b) $|N_S| = 100$



(c) $|N_S| = 200$

Fig. 4. Revenue compare between PCANM and EGNM based on the average data of twenty times running program

About the acceptance ratio, initial all of virtual networks are successfully mapped, cause by at this time, the substrate network still has all of its resource and lots of free space for virtual network to map. After a period of time, the available resource of SN is narrow down, the coming request can be denied, so not all of the virtual network are accepted. But the duration of PCANM to accept all virtual network requests is longer than EGNM. Later, the acceptance ratio of two algorithms asymptotic near to each other.

With the 200-node topologies test. For Acceptance Ratio, PCANM shows lower results in lately time of the test but if we consider about revenue chart, we can see that revenue of two algorithms is approximate each other, hence also PCANM got lower Acceptance Ratio but it accepts VN with larger revenue from that rejects lower revenue VN. EGNM is vice versa.

The main purpose of this algorithm is reducing the cost, since increase the B/C ratio. Due to mapping neighbors, this algorithm can decrease a lot of cost because it doesn't have to use more than one substrate paths to represent for a virtual link in many cases. Meanwhile, because of not considering this issue, EGNM showed less dominant than PCANM in the utilization of resources. Table 1 sums up the average of difference criteria of each algorithm.

Table 1. Average of criteria between two algorithms compare in different topologies

		EGNM	PCANM
Acceptance ratio	$\|N_S\| = 50$	0.75	0.77
	$\|N_S\| = 100$	0.81	0.81
	$\|N_S\| = 200$	0.96	0.94
Revenue/Cost	$\|N_S\| = 50$	0.54	0.69
	$\|N_S\| = 100$	0.55	0.68
	$\|N_S\| = 200$	0.55	0.69

6 Conclusion

In this paper, we proposed a totally new node mapping algorithm with 3 enhancements: sorting substrate nodes base on their degree and their available resource, mapping vertex's neighbor and checking path while taking node mapping. The algorithm maps the vertices of virtual network to the nodes of substrate network that is sorted by their degree and available resource. During the node mapping process, we examine the possibility of finding path of pairs of nodes, which improve the quality of node. Furthermore, by select the node with appropriate degree, we bring the possibility to map the neighbors of virtual nodes to the neighbors of substrate nodes. This significantly reduces the cost by mapping a link on virtual network to only one path on substrate network. Especially the list of physical nodes is created before the requests come, so it can reduces the amount of calculation.

The proposed approach (PCANM) has been compared with existing node mapping algorithms EGNM, which will be computing in time the request come. The evaluation results show that our presented algorithm gives higher performance in many important aspects. In term of revenue, PCANM is higher than EGNM about 10% in small topology (50 nodes) and 18% higher with the larger topology (200 nodes). The most significant improvement is revenue to cost ratio that is 28% better than EGNM's result and almost no change with the different data sets.

In the future work, we will try many different approaches and experiment with larger datasets in order to find the optimal solution for the problem.

References

1. Raihan Rahman, M., Aib, I., Boutaba, R.: Survivable Virtual Network Embedding. In: 9th International IFIP TC 6 Networking Conference Networking 2010, pp. 40–52 (2010)
2. Tran, H.V., Ngo, S.H.: An enhanced greedy node mapping algorithm for resource allocation in network virtualization, Journal of Science and Technology, 72–77 (2012)
3. Tran, H.V., Ngo, S.H.: Improved Heuristics for Online Node and Link Mapping Problem in Network Virtualization. In: The 13th International Conference on Computational Science and Applications (ICCSA 2013), pp.154–165 (June 2013)
4. Yu, M., Yi, Y., Rexford, J., Chiang, M.: Rethinking virtual network embedding: substrate support for path splitting and migration. SIGCOMM Comput. Commun. Rev. 38(2), pp. 17–29 (2008)
5. Razzaq, A., Hidell, M., Sjodin, P.: Virtual Network Embedding: A Hybrid Vertex Mapping Solution for Dynamic Resource Allocation. Journal of Electrical and Computer Engineering, 1–17 (2012)
6. Lischka, J., Karl, H.: A virtual network mapping algorithm based on subgraph isomorphism detection. In: The 1st ACM Workshop on Virtualized, Infrastructure Systems and Architectures, VISA 2009, p 81–88 (2009)
7. Chowdhury, N., Rahman, M.R., Boutaba, R.: Virtual network embedding with coordinated node and link mapping. In: IEEE INFOCOM 2009, pp. 783–791 (April 2009)