The Framework and Algorithms for the Survivable Mapping of Virtual Network onto a Substrate Network

Gang Sun, Hongfang Yu, Lemin Li, Vishal Anand[1] and Hao Di

School of Communication and Information Engineering, University of Electronic Science and Technology of China, Chengdu, China
[1]The College at Brockport, State University of New York, Brockport, NY, US

Abstract

Network virtualization serves as an effective method for providing a flexible and highly adaptable shared substrate network to satisfy the diversity of demands. However, the problem of efficiently mapping a virtual network onto a substrate network is intractable, as it is NP-hard. How to guarantee survivability of such a mapping efficiently is another great challenge. In this article, we investigate the survivable virtual network mapping (SVNM) problem and propose two kinds of algorithms for solving this problem efficiently. First, we formulate the model with a minimum-cost objective for the survivable network virtualization problem by mixed integer linear programming. We then devise two kinds of algorithms for solving the SVNM problem efficiently: (1) Lagrangian relaxation-based algorithms including LR-SVNM-M and LR-SVNM-D; (2) Heuristic algorithms including H-SVNM-D and H-SVNM-M. The simulation results show that these algorithms proposed in this study can be used to balance the tradeoff between time efficiency and the mapping cost.

Keywords

Mapping, Regional failure, Survivability, Virtual network.

1. Introduction

With the maturity of networking, demands on computer networks are also changing rapidly, thus a flexible and highly adaptable substrate network is needed, as the difference between the requirements can be translated into specific network requirements. Network virtualization, wherein multiple heterogeneous network architectures can share the same underlying substrate network, plays a key role in the emerging areas of cloud and grid computing [1-3]. As multiple virtual networks (VNs) now share the resources of the same underlying substrate network, efficient techniques for mapping the VNs onto the substrate network are necessary. Optical networks have become the best choice for the substrate network because of their high speed, transparent transmission, and abundant bandwidth resources [4].

Each network virtualization environment [5,6] consists of shared infrastructure and VN requests, which include a set of VN nodes; each VN node uses a certain amount of computing resource (e.g., CPU resource) of the substrate node exclusively (i.e., two or more VN nodes cannot use the resources of the same substrate node, even if there are enough resources available on that substrate node. In addition, there are also communication requirements between the VN nodes for the purpose of information and data exchange. These communication requirements constitute the edges of the VN, which are called virtual links. Therefore, another important step in network virtualization is the mapping of these virtual links onto a set of substrate links, which can provide unoccupied resources and meet the requirements of these communication demands, such as, bandwidth and delay. From the point of view of the network provider, an effective VN mapping with minimum cost will increase the utility of the substrate network and consequently produce more revenues. Furthermore, customers also expect efficient VN mapping, as it translates to reduced costs and higher efficiency in terms of delays and quality of service (QoS). In order to realize this win-win situation between customers and network providers, there exist many research studies for efficient VN mapping; however, these research studies have imposed restrictions on different dimensions of the VN mapping problem space, for example, assuming that the substrate nodes and links with infinite capacity of resources [7,8], devising distributed survivable VN mapping algorithms, without a centralized controller [9], focusing on specific types of failures [10], and ignoring the fault-tolerant ability of VN mapping against substrate network component failure[11-13].

In this article, we have studied the survivable virtual network mapping (SVNM) problem and designed a framework for solving the SVNM problem with resource constraints imposed by the substrate network. The resource capacity of a VN node and link in the framework indicate the amount of computing and bandwidth resources required by the VN request. Similarly, the capacity of the substrate nodes and links indicate the upper bound of computing resources and bandwidth they can provide, respectively. We also introduce *regional failure* in our studies. A regional failure refers to a set of substrate nodes and links, which is in the same shared risk group (SRG). Thus an active regional failure may affect the existing VN mapping. The VN mapping problem with resource constraints is intractable, as we know it to be NP-hard [5]; hence, most works devise heuristic algorithms for solving this NP-hard problem [8,14-16]. In most of these algorithms, node mapping, and link mapping are done separately. In particular, the study in [5] proposes a D-ViNE algorithm for solving the non-survivable VN mapping problem.

The framework and solutions proposed in this study address two key issues of the SVNM problem, namely, reducing VN mapping cost, and reducing the computational complexity of the VN mapping algorithm. It should be noted that these two issues are complementary, for example, solutions that help reduce VN mapping costs typically incur higher computational costs and time. In this article, we first model the SVNM problem with resource constraints using the mixed integer linear programming (MILP) approach for achieving the optimal solution of the SVNM problem. We extend our study on the basis of the MILP model, and propose two kinds of algorithms for solving the SVNM problem: (1) Lagrangian relaxation [17]-based algorithms including LR-SVNM-D and LR-SVNM-M and (2) Heuristic algorithms including H-SVNM-D and H-SVNM-M. The main aim of these two algorithms is to decompose the primal NP-hard problem into several sub-problems in order to reduce the computational complexity at the expense of increasing the total VN mapping cost. Simulation results show that the algorithms proposed in this study can balance the tradeoff between VN mapping cost and the computational complexity of the VN mapping algorithm.

We organize the rest of this article as follows. In section 2, we provide a statement of the survivable VN mapping problem and formulate the MILP model. We describe decomposition of the MILP model and give detailed algorithms for SVNM in section 3. The simulation environment to evaluate the performance of our approaches and associated experiment results are given in section 4. Section 5 concludes the article.

2. Problem Statement and MILP Formulation

In this article, we reuse the framework proposed in our earlier article [16] to describe the VN mapping problem and MILP model.

2.1 Substrate Network

On account of their advantage such as high speed, high signal-to-noise ratio (SNR), transparent transmission, and abundant bandwidth resources [4], the optical WDM mesh networks are playing a major role as substrate networks.

We model the substrate network as an undirected graph $G_S = (V_S, E_S)$, where V_S is the set of substrate nodes, and E_S represents the set of bidirectional fiber links. Note that we suppose that all the substrate nodes are facility nodes (with computing resource) in this article, namely $V_F = V_S$.

Each facility node $v_F \in V_F$ can provide computing resources and $c(v_F)$ represents the available capacity of the computing resources at facility node v_F. The cost of the per unit computing resource on facility node v_F is $cf(v_F)$. For each link, $e_S \in E_S$, the available bandwidth capacity is $b(e_s)$ and the cost of per unit bandwidth capacity is $cl(e_s)$.

Figure 1 (a) shows a substrate network, where the numbers aside the links represent the available bandwidth and the cost of per unit bandwidth capacity and the numbers in the rectangles represent the available computing resources and cost of per unit computing resource.

2.2 VN Request

A VN request with QoS requirements is modeled as a weighted undirected graph $G_L = (V_L, E_L)$, where V_L rep-

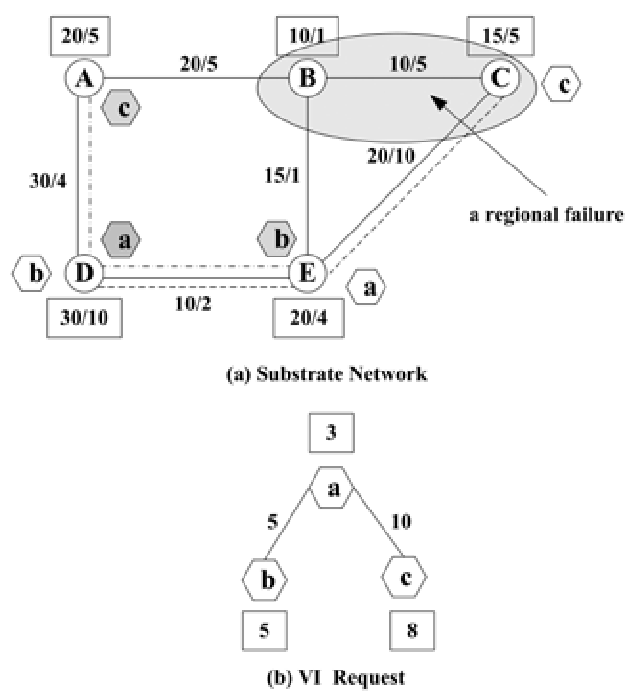

Figure 1: Mapping a VN request onto a substrate network.

resents the set of VN nodes, and E_L represents the set of bidirectional communication demands among the VN nodes.

Each VN node, $v_L \in V_L$, needs a certain amount of computing resources for executing the applications, denoted by $\varepsilon(v_L)$. Furthermore, each communication demand, $e_L \in E_L$, with a bandwidth requirement for information and data exchanging, is denoted by $b(e_L)$. Figure 1 (b) shows a VN request with three VN nodes and two VN links, and the associated computing and communication resource requirements.

2.3 Regional Failures

The authors in [18] discuss the scenario that there exist only a finite number of distinct regional failures in a given geographical area. Accordingly, we assume a set of possible regional failures, R, which is given in advance in our studies, and there is at most one of these regional failures that is active at any time. One or more substrate nodes and relative substrate links will be affected simultaneously when a regional failure $r_i \in R$ $(i = 0,1,2.....|R|-1)$ turns from inactive to active. In fact we can generalize a regional failure to the concept of shared risk group (SRG) failures. Note that, we use r_0 to denote a scenario in which no failure occurs, in our studies.

2.4 Survivable VN Mapping Problem

Mapping a VN request G_L over a substrate network G_S with survivability, against any single regional failure r consists of five parts: Initial VN request mapping, redundant facility node allocation, redundant link allocation, VN request checkpointing, and VN request migration.

Initially, without the consideration of possible failures, we need to allocate a separate facility node and the required computing resources for each VN node of the VN request, as well as, set up paths and a reserve required bandwidth to transmit data or information among the VN nodes. Assuming that the number of VN nodes is *Num*, then the exact *Num* facility nodes are occupied in this initial request mapping.

In our failure-dependent SVNM approach, we assume that if a facility node initially occupied by a VN node is within the failed region, we then need one backup facility node, outside the failed region, in order to remap the VN node. Note that the choice of the additional backup facility node(s) depends on the active failure $r_i \in R$, and may differ from one failure to another. To ensure that such a backup facility node can be found in the event of a regional failure, we reserve the backup facility node in advance. As a regional failure may affect multiple facility nodes allocated for the initial VN request mapping, a sufficient set of redundant facility nodes have to be pre-allocated.

Similarly, for each regional failure, $r \in R$, we need to pre-allocate enough redundant bandwidth on the substrate links, which do not belong to regional failure r, to support the communication between the VN nodes under regional failure r. In particularly, if a VN node is to be restored at a pre-allocated backup facility node, we must set up new available paths between that backup facility node and the other surviving facility nodes.

To eliminate the need for starting over the job/computation of a VN node when it is disrupted by a failure, we need to checkpoint the VN node. More specifically, we periodically send the status of the VN node (e.g., virtual machine image) to the backup facility node such that the VN request can be restored from a previous saved state when a regional failure occurs. Finally, we need to perform VN request migration, that is, migrate the failed facility nodes to the backup facility nodes.

In Figure 1, the VN nodes a, b, and c are initially (before failure) mapped to the facility nodes *E*, *D*, and *C* using the bandwidth on substrate link *E-C* and *E-D*, respectively. The regional failure destroys facility nodes *B* and *C*, and the substrate link *B-C*. The substrate network now remaps the VN request such that the VN node *a* is now mapped onto facility node *D*, *b* is mapped onto facility node *E*, and *c* is mapped onto facility node *A* using the bandwidth on substrate links *D-A* and *D-E*, and the jobs on these failed facility nodes are migrated to the newly mapped facility nodes.

2.5 Problem Description

Given: A substrate network $G_S = (V_S,E_S)$, a VN request $G_L = (V_L,E_L)$, a set of possible regional failures R.

Problem: Jointly allocate computing and link capacity resources of the substrate network, including the backup resources, such that the total cost of the survivable mapping of the VN request, that is, the sum of computation and communication cost is minimized.

As we assume that no more than one regional failure occurs at any one time, the resources reserved on the facility nodes and fiber links can be shared among different failure scenarios. Accordingly to ensure survivability under any regional failure, the total computing resources that are allocated on substrate node n, denoted by rn_n, is the maximum of all computing resources under any failure. Similarly, the total bandwidth allocated on substrate link e, denoted by rl_e, is the maximum of all bandwidth resources under any failure.

2.6 MILP Formulation for SVNM Problem

In this article, we use $Sc(e)$ and $Dt(e)$ to denote source and sink node of link$e$, respectively. To formulate the MILP model for an SVNM problem, we apply the following graph transformation similar to [5], to G_S. We add $|V_L|$ virtual nodes into G_S; each virtual node corresponds to a VN node in the VN request. Each virtual node is set with an infinite computing capacity. Each virtual node v_L is connected to all the facility nodes, $v_F \in V_F$, which have enough available computing resources required by the corresponding VN node v_L. We call the links connecting virtual nodes and facility nodes as virtual edges. We assume that the bandwidth resources on each virtual edge are unlimited. Each virtual node v_L is unaffected by any regional failure in R. However, for each virtual edge connecting virtual node v_L and facility node v_F, if v_F is within a region $r \in R$, then that virtual edge is also inside the region r. By means of graph transformation, we achieve the following augmented graph G^* as $G^* = (V^*, E^*)$, where $V^* = V_s \cup V_L$, and $E^* = E_S \cup \{e \mid Sc(e) \in V_F, Dt(e) \in V_L\} \cup \{e \mid Sc(e) \in V_L, Dt(e) \in V_F\}$.

Based on the above-mentioned graph transformation, the SVNM problem can be formulated as a mixed integer Multi-Commodity Flow (MCF) problem, in which a communication demand d in G_L is considered as a commodity. We present the detailed MILP formulation for the SVNM problem as follows: Table 1 lists the key notations used in the MILP model.

Table 1: Summary of key notations

Notation	Meaning
$Sc(e)$	The source node of link e
$Dt(e)$	The destination node of link e
fl_e	Total bandwidth required on substrate link e.
rn_n	Total computing resources required on facility node n
$rl_{e,r}$	Total bandwidth required on substrate link e under regional failure r.
$rn_{n,r}$	Total computing resource required on facility node n under regional failure r.
$\chi_e^{d,r}$	The used bandwidth on substrate link e by demand d under regional failure r
α_e^r	Whether link e is used under regional failure r. 1 if is used, and 0 otherwise
b_e	Total bandwidth capacity of link e
cl_e	The cost of per unit bandwidth of substrate link e
$c(n)$	The computing resources available on facility node n
cn_n	The cost of per unit computing resource of facility node n
$\varepsilon(n)$	The computing resources required by virtual node n
D	The set of communication demands of G_L
b_d	The bandwidth required by communication demand d
s_d, t_d	The source and sink node of communication demand d
V^*, E^*	The set of nodes and links of augmented graph G^*
β_e^r	Whether link e is included in regional failure r
η_m^r	Whether node m is included in regional failure r
T	Big constant, which equals to *10,000* in this article

Objective function

$$min \sum_{e \in E_s} fl_e \times cl_e + \sum_{m \in V_s} rn_m \times cn_m \tag{1}$$

The objective function in (1) minimizes the total mapping cost (cost of CPU and bandwidth resources). It can guarantee that the solution is achieved for the SVNM problem with minimum total cost.

Constraints

$$\sum_{d \in D} (\chi_e^{d,r} + \chi_e^{d,r}) \le 2 \times b_e \times \alpha_e^r, \quad \forall e \in E^*, \forall r \in R \tag{2}$$

$$\alpha_e^r \times \varepsilon(Sc(e)) \le c(Dt(e)), \forall e \in E_L^*, \quad Sc(e) \in V_L, Dt(e) \in V_S, \forall r \in R \tag{3}$$

Constraints (2) and (3) are the link and node capacity constraints; they ensure that the required resources must neither exceed link bandwidth capacity nor the node computing resource capacity.

$$\sum_{\substack{e \in E^*,\\ Sc(e)=m}} \chi_e^{d,r} - \sum_{\substack{e \in E^*,\\ Dt(e)=m}} \chi_e^{d,r} = 0, \forall d \in D, \forall m \in V^* \backslash \{s_d, t_d\}, \forall r \in R \tag{4}$$

$$\sum_{e \in E^*, Sc(e)=s_d} \chi_e^{d,r} - \sum_{e \in E^*, Dt(e)=s_d} \chi_e^{d,r} = b_d, \forall d \in D, \forall r \in R \tag{5}$$

$$\sum_{e \in E^*, Dt(e)=t_d} \chi_e^{d,r} - \sum_{e \in E^*, Sc(e)=t_d} \chi_e^{d,r} = b_d, \quad \forall d \in D, \forall r \in R \tag{6}$$

$$\sum_{e \in E^*, Dt(e)=s_d} \chi_e^{d,r} = 0, \quad \forall d \in D, \forall r \in R \tag{7}$$

$$\sum_{e \in E^*, Sc(e)=t_d} \chi_e^{d,r} = 0, \quad \forall d \in D, \forall r \in R \tag{8}$$

Equations (4)-(8) are flow conservation constraints, these flow conservations can be guaranteed not only on substrate nodes, but also on virtual nodes.

$$\alpha_e^r \le 1 - \beta_e^r, \quad \forall e \in E^*, \forall r \in R \tag{9}$$

$$\sum_{e \in E^*, Sc(e)=m} \alpha_e^r \le T \times (1 - \eta_m^r), \quad \forall m \in V_S, \forall r \in R \tag{10}$$

Constraints (9) and (10) ensure that only when a link or node is available under regional failure r, then assigning the link or node under regional failure$r$ is valid.

$$\sum_{e\in E^*, Sc(e)=m, Dt(e)\in V_S} \alpha_e^r = 1, \quad \forall m\in V_L, \forall r\in R \tag{11}$$

$$\sum_{e\in E^*, Sc(e)\in V_L, Dt(e)=n} \alpha_e^r \le 1-\eta_n^r, \quad \forall n\in V_S, \forall r\in R \tag{12}$$

Equation (11) ensures that only one substrate node is selected for mapping the virtual node under any regional failure r. Constraint (12) ensures that no more than one virtual node is mapped onto an available substrate node under any failure r; and when the substrate node is unavailable under failure r, there should be no virtual node mapped onto it.

$$\alpha_{e_1}^r=\alpha_{e_2}^r, \forall r\in R, \forall e_1\in E^*, Sc(e_2)=Dt(e_1), Dt(e_2)=Sc(e_1) \tag{13}$$

$$rl_{e,r} = \sum_{d\in D}\chi_e^{d,r} \le 2\times b_e\times \alpha_e^r, \quad \forall e\in E_s, \forall r\in R \tag{14}$$

$$rn_{n,r} = \sum_{e\in E^*, Sc(e)\in V_L, Dt(e)=n} \alpha_{mn}^r \times \varepsilon(m), \quad \forall n\in V_s, \forall r\in R \tag{15}$$

$$rl_{e,r} \le fl_e, \quad \forall e\in E_s, \forall r\in R \tag{16}$$

$$rn_{n,r} \le rn_n, \quad \forall n\in V_s, \forall r\in R \tag{17}$$

Constraint (13) denotes that the binary variables for the two links in the opposite direction, between any node pair, have the same value. Equations (14) and (15) denote the total amount of resources required on each substrate link and node under regional failure r, respectively. Constraints (16) and (17) ensure that the amount of resources required on each substrate link or node, under any regional failure r, must not exceed the total amount of resources required on each substrate link or node.

3. Algorithms for SVNM Problem

The survivable virtual network mapping problem with link and node capacity constraints can be formulated as an MILP problem, which is described in Section II. However, this MILP problem is intractable, as it is NP-hard. Consequently, we propose the Lagrangian relaxation [17]-based algorithm (LR-SVNM) and decomposition technique based heuristic algorithm (H-SVNM) in this section, in order to reduce the computational complexity and solve the problem efficiently. The main idea of our algorithms is to decompose the primal problem into $|R|$ sub-problems and solve these sub-problems separately, thereby, reducing the computational complexity and enhancing the time efficiency.

3.1 Heuristic Algorithm for SVNM (H-SVNM)

D-ViNE algorithm with good performance on computation time for non-survivable VN mapping is proposed in [5]. In this algorithm the mapping for virtual nodes and links are completed in ordered phases. The SOUM* and IOCM* algorithms proposed in [14] can recover from failures. In the SOUM* algorithm, the SVNM problem can be decomposed into $|R|$ separate non-survivable virtual network mapping (NSVNM) problems, for reducing the computational complexity to some extent. However, the performance in terms of mapping cost is not as good, and there is room for improvement.

In this section, we incorporate the advantages of D-ViNE and SOUM* algorithms and propose a heuristic algorithm for the SVNM problem, called H-SVNM. In the H-SVNM algorithm, the primal problem is also decomposed into $|R|$ sub-problems. We may solve each sub-problem by using the improved D-ViNE algorithm, called D-ViNE* [Figure 2], in which we compute p_{z_aver} and Z_{min} for each VN node $v_L \in V_L$ as $p_{z\_aver} = average\{p_Z \,|\varphi(z)=0\}$ and $Z_{min} = \arg\, min\, \{|p_z - p_{z\_aver}|, \varphi(z)=0, z\in\Omega(v_L)\}$ (H-SVNM-D); or each sub-problem will be solved by MILP (H-SVNM-M). Consequently, we achieve node mapping, as $EN(v_L)\leftarrow Z_{min}$. We can get a set of mapping solutions for all the sub-problems, $E=\{E_i\,|i=0,1,2,....|R|-1\}$. As a mapping solution E_i for failure r_i may also be used for recovering from other failures, there may be redundancies in the mapping solution set E. We use the greedy min-cost set cover algorithm [16] to eliminate the redundant mappings in E. The framework of H-SVNM is shown in Figure 3. Details of the H-SVNM algorithm are shown in Figure 4.

```
Initialization. Create augmented substrate graph G*
Solve VNE_LP_RELAX
for all v_S ∈ V_S do
φ(v_S) ← 0
end for
Initial Connected-Substrate-Node-Set Θ
for all v_L ∈ V_L do
if Ω(v_L) ∩ {v_S ∈ V_S | φ(v_S) = 1} = ∅ then
VN request cannot be satisfied
return
end if
for all z ∈ Ω(v_L) do
p_z ← Σ_i (f^i_{μ(v_L)z} + f^i_{zμ(v_L)}) χ_{μ(v_L)z}
end for
Compute p_{z_aver} as: p_{z_aver} = average{p_z | φ(z)=0, z∈Θ}
Let z_min = arg min{|p_z − p_{z_aver}| φ(z)=0, z∈Θ, z∈Ω(v_L)}
Set M_N(v_L) ← z_min
φ(z_min) ← 1
end for
Solve MCF to map virtual edges.
```

Figure 2: The D-ViNE* algorithm.

3.2 Lagrangian Relaxation-based Algorithm for SVNM (LR-SVNM)

As the original MILP problem is concave, we can use the Lagrangian Relaxation approach to solve it. Our main idea is: Decomposing the NP-hard problem into several sub-problems by relaxing certain constraints. Each sub-problem may also be solved by MILP (LR-SVNM-M), or each sub-problem may be solved by using D-ViNE* (LR-SVNM-D). The framework of LR-SVNM is shown in Figure 5.

In the LR-SVNM algorithm, we chose constraints (16) and (17) of the original MILP problem for relaxing. Here, we denote the Lagrange multiplier as λ and θ (where $\lambda \geq 0$, $\theta \geq 0$), then the Lagrangian function can be written by relaxing the coupling constraint (16) and (17) as:

$$\begin{aligned} L(\lambda,\theta) =& \sum_{e\in E_s} fl_e \times cl_e + \sum_{m\in V_s} rn_m \times cn_m \\ &+ \sum_{e\in E_S}\sum_{r\in R} \lambda_{e,r} \times (rl_{e,r} - fl_e) + \sum_{n\in V_S}\sum_{r\in R} \theta_{n,r} \times (rn_{n,r} - rn_n) \\ =& \sum_{e\in E_S} fl_e \times (cl_e - \sum_{r\in R}\lambda_{e,r}) + \sum_{n\in V_S} rn_n \times (cn_n - \sum_{r\in R}\theta_{n,r}) \\ &+ \sum_{r\in R}(\sum_{e\in E_S}\lambda_{e,r} \times rl_{e,r} + \sum_{n\in V_S}\theta_{n,r} \times rn_{n,r}) \end{aligned} \tag{18}$$

In equation (18), variables fl_e , rn_n , $rl_{e,r}$ and $rn_{n,r}$ are called primal variables and the Lagrange multipliers λ and θ are dual variables. Similarly, the objective function (1) of the original MILP problem is primal objective, whereas, the minimum value of the Lagrange dual problem (18) is called dual objective and si denoted by $\Gamma(\lambda,\theta)$. Then $\Gamma(\lambda,\theta)$ can be formulated as follows:

$$\Gamma(\lambda,\theta) = \inf_{\lambda,\theta} L(\lambda,\theta). \tag{19}$$

Since the dual function of $\Gamma(\lambda,\theta)$ is concave, it can then be maximized to obtain a lower bound on the optimal value of the original MILP problem. We can formulate the dual problem of the original MILP problem as:

$$\begin{aligned} &max \;\; \Gamma(\lambda,\theta) \\ &s.t. \;\; \lambda \geq 0, \theta \geq 0. \end{aligned} \tag{20}$$

Obviously, the original MILP problem has been separated into $|R|$ independent sub-problems for solving by Lagrangian (18). We can see clearly that each independent problem can be solved separately by solving the following two sub-problems (21) and (22):

$$\begin{aligned} &min \sum_{e\in E_S} fl_e \times (cl_e - \sum_{r\in R}\lambda_{e,r}) + \sum_{n\in V_S} rn_n \times (cn_n - \sum_{r\in R}\theta_{n,r}) \\ &s.t. \;\; fl \geq 0, rn \geq 0. \end{aligned} \tag{21}$$

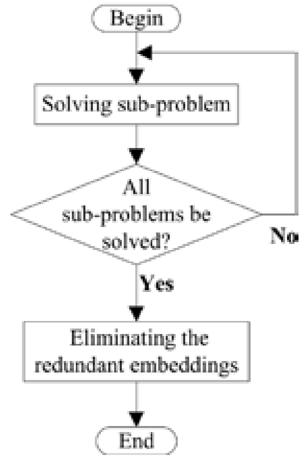

Figure 3: The framework of H-SVNM algorithm.

1: for all $r_i \in R \;\; (i = 0....|R|-1)$ do

2: $G^* \leftarrow G^* - \{(V_f, E_f) | V_f \in r_i, E_f \in r_i\}$

3: Solving each sub-problem by using MILP or D-ViNE* algorithm to get embedding E_i for failure r_i

4: end for

5: Call greedy min-cost set cover algorithm [16] to eliminate the redundant embeddings to obtain E_{best}.

Figure 4: The H-SVNM algorithm.

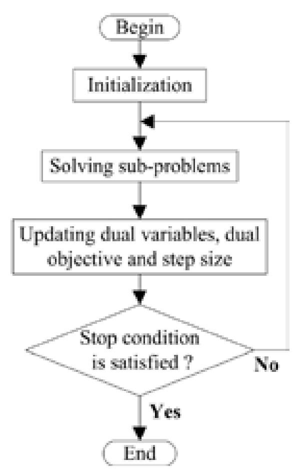

Figure 5: The framework of the LR-SVNM algorithm.

$$min \sum_{r\in R}(\sum_{e\in E_S}\lambda_{e,r} \times rl_{e,r} + \sum_{n\in V_S}\theta_{n,r} \times rn_{n,r}) \tag{22}$$

s.t. constraints(2)-(15).

Note that the values of λ and θ in problems (21) and (22) are the solutions of the dual problem (20).

We solved the dual problem (20) by using the steepest descent approach in this article, for updating λ and θ after solving problems (21) and (22) as:

$$\lambda_{e,r}(k+1)=\left[\lambda_{e,r}(k)+t1_k\times(rl_{e,r}(k)-fl_e(k))\right]^+,\quad \forall e\in E_S, r\in R \tag{23}$$

$$\theta_{n,r}(k+1)=\left[\theta_{n,r}(k)+t2_k\times(rn_{n,r}(k)-rn_n(k))\right]^+,\quad \forall n\in V_S, r\in R \tag{24}$$

Where $rl_{e,r}$, fl_e, $rn_{n,r}$ and rn_n are solutions of problems (21) and (22). $t1_k$ and $t2_k$ are step sizes for the k^{th} iteration, and can be computed as follows:

$$t1_k=\tau\times(f^*-\Gamma(\lambda(k),\theta(k)))/\sum_{e\in E_S}\sum_{r\in R}(rl_{e,r}(k)-fl_e(k))^2 \tag{25}$$

$$t2_k=\tau\times(f^*-\Gamma(\lambda(k),\theta(k)))/\sum_{n\in V_S}\sum_{r\in R}(rn_{n,r}(k)-rn_n(k))^2 \tag{26}$$

Where f^* denotes the upper bound of dual objective value, and is easy to achieve, as any feasible solution for a primal problem is an upper bound for a dual objective.

The iteration for solving all the above-mentioned problems do not stop until a predefined stop condition is satisfied. The dual variables $\lambda(k)$ and $\theta(k)$ will converge to the optimal values λ^* and θ^*, while $k\to\infty$ and the primal variables also converge to their optimal values, as the duality gap for the primal problem is zero and the solutions for (21) and (22) are stable [19].

Summarizing, we give detailed LR-SVNM algorithm for the SVNM problem in Figure 6.

1: Initialization. Set $k_{\max}$, $\tau_{iter\_\max}$ $\lambda(0)$, $\theta(0)$, $k=0$, $\tau=1.0$, $\tau_{\min}=0.005$, $\tau_{iter}=0$, $f_{best}=\infty$.

2: Set $\tau_{iter}=\tau_{iter}+1$; Solve problem (21) and (22) to achieve solutions $rl_{e,r}(k)$, $rn_{n,r}(k)$, $fl_e^{\wedge}(k)$, $rn_n^{\wedge}(k)$ by using the given value of $\lambda(k)$ and $\theta(k)$.

3: Use solution of problem (21) and (22) to compute:
- Compute feasible $fl_e(k)$ and $rn_n(k)$ as:
  $fl_e(k)\leftarrow \underset{r\in R}{Max}\, rl_{e,r}(k)$, $rn_n(k)\leftarrow \underset{r\in R}{Max}\, rn_{n,r}(k)$;
- Use $fl_e(k)$ and $rn_n(k)$ to compute the objective value f of the primal problem;
- If $f < f_{best}$, then: $f_{best}=f$, $fl_e^{best}(k)=fl_e(k)$, $rn_n^{best}(k)=rn_n(k)$, $f^*=f_{best}$;
- If $t_{iter}>t_{iter\_max}$, then $\tau=max\{\tau\times 0.618,\tau_{\min}\}$ $\tau_{iter}=0$.

4: Use solution of problem (21) and (22) to update:
- Dual objective $\Gamma(\lambda(k),\theta(k))$ (refer to eq. (18) – (19));
- Dual variables λ and θ (refer to eq. (23) – (24));
- Step size $t1_k$ and $t2_k$ (refer to eq. (25) – (26)).

5: Set $k=k+1$.

6: Go to Step 2 until $k>k_{\max}$.

Figure 6: The LR-SVNM algorithm.

4. Simulation Experiment Results

We evaluated the performance of our proposed framework and algorithms using detailed simulation experiments. In this section, a detailed description for a simulation environment is given and the main experimental results are presented.

4.1 Simulation Environment

To evaluate the effectiveness of our approaches we conducted simulation studies on four different network topologies, as shown in Figure 7. The CERNET, NSFNET, and USANET are well-known network topologies that have traditionally been used for conducting network and simulation experiments. The four network topologies chosen vary in terms of the number of nodes, links, and connectivity, and thus provide a good basis for the evaluation of our approaches. In these four substrate networks, all link bandwidth capacities and node resource capacities are assumed to be 50 units. We assume per unit node resource cost and per unit link bandwidth cost to be equal to one unit. In Net-1, Net-2, and Net-3, we randomly choose to fail two substrate nodes when a regional failure occurs; and in Net-4, we randomly fail three substrate nodes when a regional failure occurs. There is, at the most, one active regional failure r at any given time.

The virtual networks or VN requests are generated randomly, such that the number of VN nodes is equal to a given number N and the average degree of connectivity of the VN request is about 2.5. We assume that each VN node requires one unit computing resource capacity, and one unit of bandwidth resources is required by each of the communication demands between the VN nodes.

we have implemented our algorithms by using Visual Studio 2005 and C++ programming language. In our simulations, each MILP problem is solved by the CPLEX solver.

4.2 Algorithms Used for Comparison

In our experiments, we have simulated and compared the performances of the following algorithms: IOCM* [5], SOUM* [5], LR-SVNM-D, LR-SVNM-M, H-SVNM-D, and H-SVNM-M, which use different VN mapping strategies. In addition we have also compared the per-

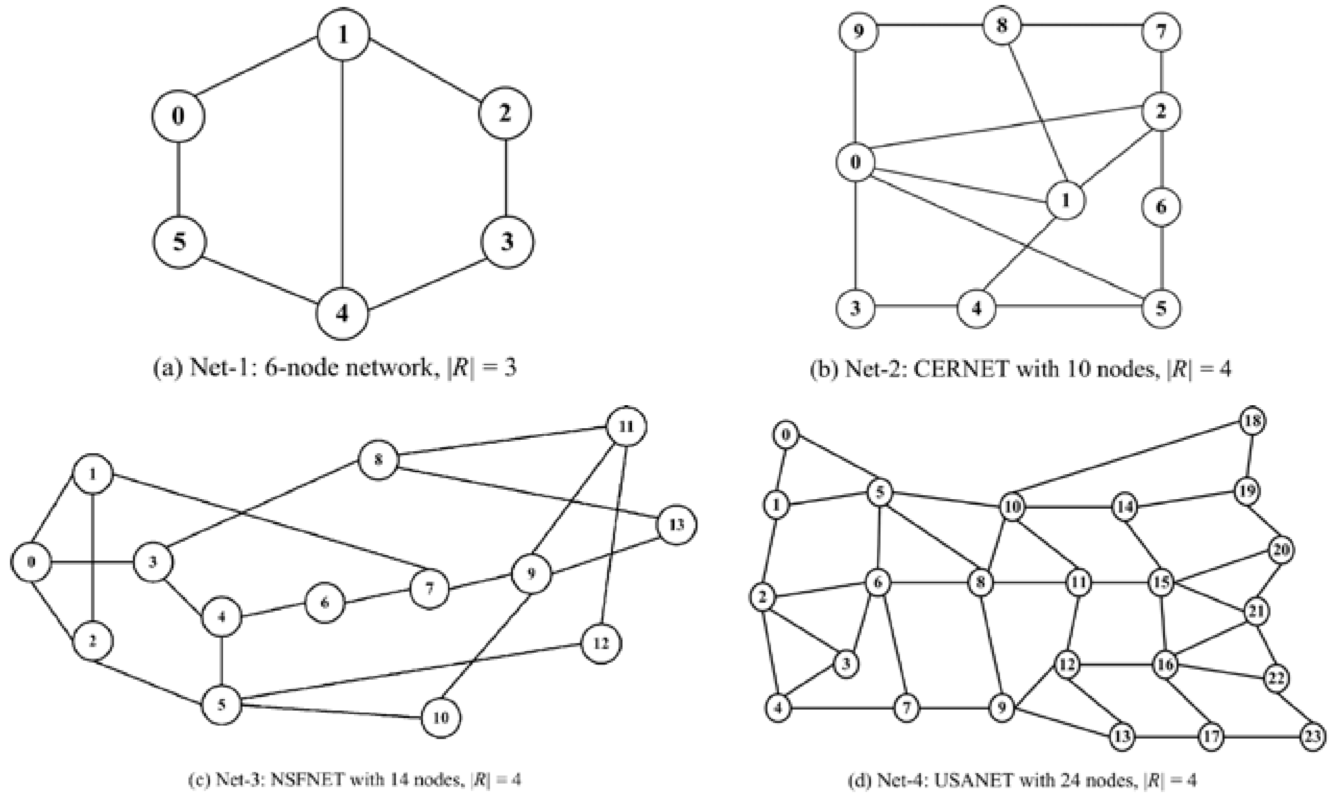

Figure 7: Substrate networks used for simulation experiments.

formance of these algorithms with the optimal results obtained from the MILP.

Table 2 enumerates the notations used for each of these algorithms.

4.3 Performance Metrics

As the two key issues of the SVNM problem algorithm are VN mapping cost and computational complexity (as mentioned in Section 1). We use the following two metrics to evaluate the performance of the algorithms mentioned in Table 2.

(1) Total Mapping Cost: This is the total cost of reserving substrate network resources to tolerate any failure scenario *r*. It is the sum of the computing cost on all facility nodes and the bandwidth cost on all substrate links.
(2) Time Efficiency: This is the time consumed by an algorithm for complementing the VN mapping.

4.4 Simulation Experiment Results

We have compared the performance of different algorithms in terms of total cost and time efficiency of VN mapping under various substrate network topologies and with different VN requests, that is, different number of VN nodes and links in the request. We use different notations in our simulation results to indicate

Table 2: Algorithms compared

Notation	Description of algorithm
IOCM*	The algorithm proposed in [5]
SOUM*	The algorithm proposed in [5]
LR-SVNM-M	Lagrangian Relaxation-based algorithm for an SVNM problem, each sub-problem is solved by MILP
LR-SVNM-D	Lagrangian Relaxation-based algorithm for an SVNM problem, each sub-problem is solved by D-ViNE*
H-SVNM-D	Heuristic algorithm for an SVNM problem, each sub-problem is solved by D-ViNE*
H-SVNM-M	Heuristic algorithm for an SVNM problem, each sub-problem is solved by MILP
MILP	Solve an SVNM problem by Mixed Integer Linear Programming

*name of algorithm

the different substrate networks and VN requests that have been used in our experiments. For example, the notation '6-2' denotes '6 substrate nodes and 2 VN nodes'. In our simulation studies, we use Net-1, Net-2, Net-3, and Net-4, and the number of VN nodes varies from 2 to 18.

(1) The convergence of LR-SVNM Algorithm: We set $\lambda(0) = 1$ and $\theta(0) = 3$, in this set of experimental simulations. In Figure 8, the experimental results show that the Lagrangian relaxation-based algorithms LR-SVNM can guarantee a solution for a primal problem,

and furthermore converge to a stable one by solving the sub-problems iteratively.

(2) Comparison of total mapping cost: Figure 9 shows the simulation results that compare the total mapping costs of the various algorithms. We make the following conclusions from the simulation results:

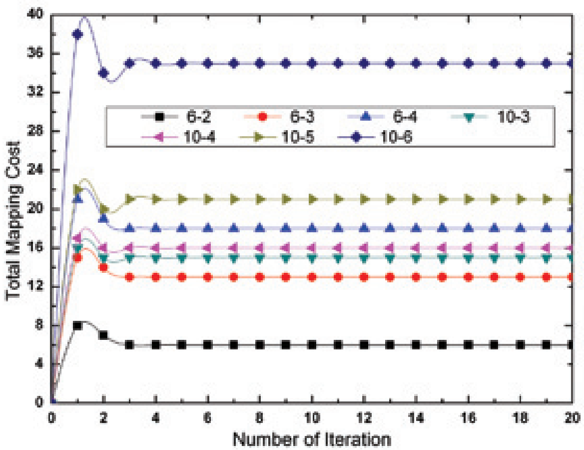

Figure 8: Convergence of the LR-SVNM algorithm.

The VN mapping cost of our algorithms is lower than IOCM* and SOUM* proposed in [5]. Under small size substrate network (Net-1), heuristic algorithms (H-SVNM and H-SVNM-M) and Lagrangian relaxation-based algorithms (LR-SVNM-M and LR-SVNM-D), result in a similar mapping cost, which is similar (or equal) to that of the MILP. However, when the substrate network size is large [Figure 9 (b)], H-SVNM-M achieves a lower mapping cost than H-SVNM-D and LR-SVNM-M, with a better mapping cost than LR-SVNM-D, as the former

Table 3: Comparison of time efficiency (1)

Time (Sec.)	Node number						
	6 – 2	6 – 3	6 – 4	10 – 3	10 – 4	10 – 5	10 – 6
IOCM*	0.01	0.01	0.04	0.1	0.12	0.35	1.05
SOUM*	0.01	0.01	0.01	0.03	0.1	0.2	0.45
H-SVNM-D	0.4	0.5	0.7	1.1	1.4	1.8	2.6
H-SVNM-M	0.5	0.9	1.9	3.1	23.6	94.5	3126.8
LR-SVNM-M	1.6	3.0	6.4	12.4	83.4	407.3	11345.1
LR-SVNM-D	1.5	1.9	2.0	3.2	4.2	5.3	8.1
MILP	0.3	3.1	6.1	108.1	790514.4	829713.8	875501.5

*name of algorithm

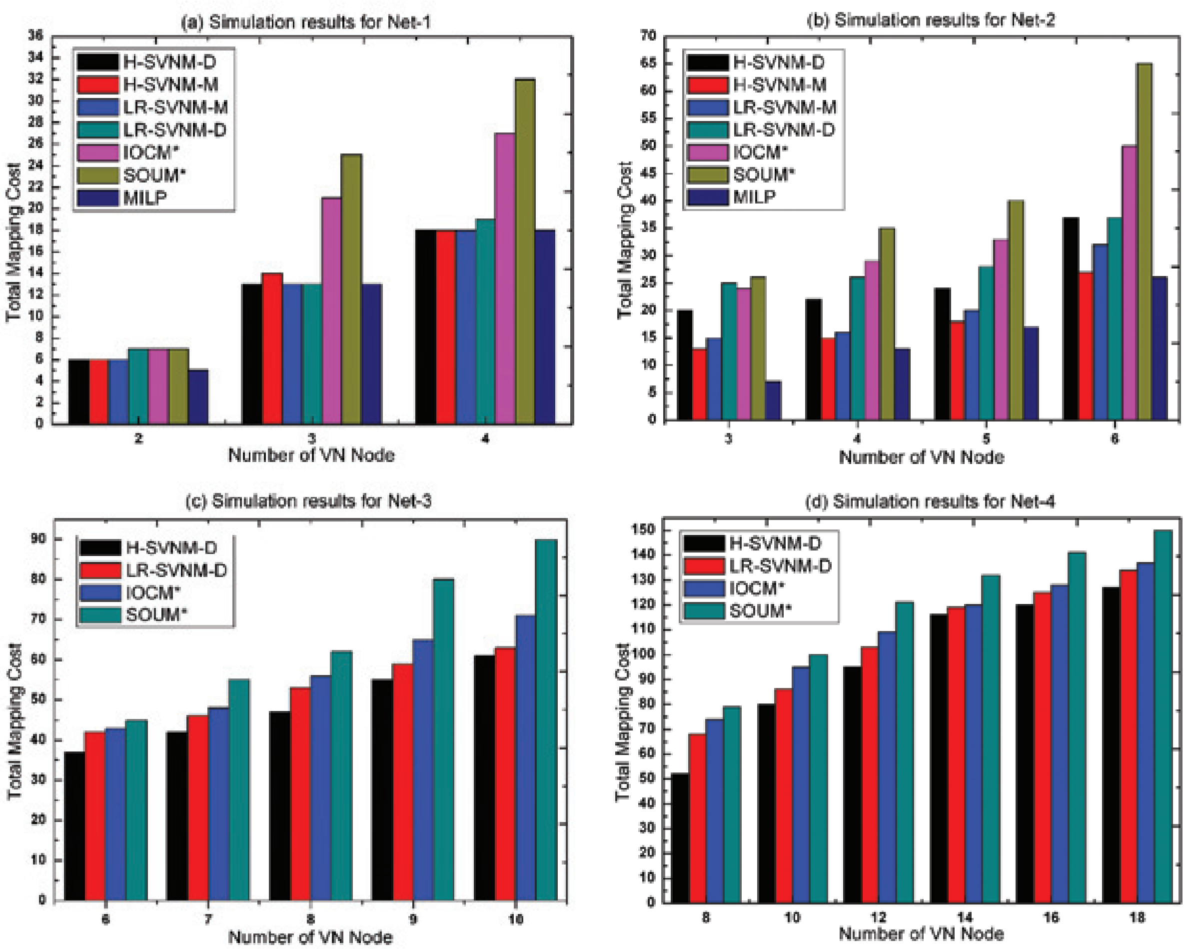

Figure 9: Comparison of total mapping cost.

Table 4: Comparison of time efficiency (2)

Time (Sec.)	Node number										
	14 – 6	14 – 7	14 – 8	14 – 9	14 – 10	24 – 8	24 – 10	24 – 12	24 – 14	24 – 16	24 – 18
IOCM*	1.02	1.2	2.05	2.56	2.67	10	11	11	13	14	12
SOUM*	0.2	0.32	0.38	0.48	0.82	1.14	1.68	2.01	2.23	2.45	2.9
H-SVNM-D	5	6	8	10	13	20	34	50	78	107	155
LR-SVNM-D	65	52	22	30	13	448	107	627	1855	2053	1393

*name of algorithm

solves each sub-problem by using the MILP, while the latter employs heuristic-based algorithms for solving the sub-problems. H-SVNM-M has a better performance than LR-SVNM-M in terms of mapping cost, as the Lagrangian Relaxation-based algorithm reduces the computational complexity at the cost of losing some feasible solutions to the original problem.

(3) Comparison of time efficiency: From Tables 3 and 4 we can see clearly that the time efficiency of our algorithms has been enhanced considerably compared to that of MILP. However, it is not as good as that of the IOCM* and SOUM* algorithms. In addition we note that H-SVNM-M has a performance that is worse than that of H-SVNM-D, and the time efficiency of LR-SVNM-M is not as good as that of LR-SVNM-D. The reason for this is that each sub-problem is solved by MILP in H-SVNM-M and LR-SVNM-M, while in H-SVNM-D and LR-SVNM-D all sub-problems are solved by using the heuristic algorithm D-ViNE*. However, we also note that heuristic algorithms H-SVNM-D and H-SVNM-M perform better than the Lagrangian relaxation-based algorithms LR-SVNM-M and LR-SVNM-D in terms of computational time, as the latter have to solve the sub-problems iteratively.

Lagrangian Relaxation-based algorithms reduce computational complexity efficiently, enhance time efficiency considerably, and guarantee the solution of the original problem to converge to a stable one. However, their performance in terms of mapping cost and time efficiency is inferior to that of the heuristic algorithms, due to their inherent disadvantages such as iterative computation and loss of some feasible solutions. We can achieve a total mapping cost that is similar to the optimal value by using MILP for solving each sub-problem.

5. Conclusion

Network virtualization serves as an effective method for providing a flexible and highly adaptable shared substrate network, for satisfying a diversity of demands. Thus, in such a virtualized environment the problem of efficiently mapping a virtual network onto a shared substrate network becomes important. As many virtual networks now share the resources of a common substrate network, guaranteeing the survivability of such a mapping efficiently is increasingly important and challenging. We have formulated the survivable virtual network mapping problem with resource constraints such as the MILP problem. However, this MILP problem is intractable as it is NP-hard. We have proposed two kinds of algorithms for solving this problem efficiently: Lagrangian Relaxation-based algorithms and heuristic-based algorithms. In our algorithms, we decompose the primal problem into several sub-problems to reduce the complexity of computation and to enhance time efficiency. We compare the effectiveness of our approaches using detailed simulation under various substrate networks and virtual network topologies. Our results show that our algorithms perform well and can provide a balanced tradeoff between time complexity and mapping costs.

In our future research study, we propose to design online adaptive algorithms for the mapping of multiple virtual networks simultaneously considering random failure.

6. Acknowledgment

This research was partially supported by Natural Science Foundation of China grant (No.60872032, 60972030 and 61001084), the National Grand Fundamental Research 973 Program of China under Grant No.2007CB307104, and the Fundamental Research Funds for the Central Universities (ZYGX2010J002).

References

1. T. Anderson, L. Peterson, S. Shenker, and J. Turner, “Overcoming the Internet impasse through virtualization,” Computer. vol. 38. No. 4. pp. 34-41, 2005.
2. I. Foster, Y. Zhao, I. Raicu, and S. Lu, “Cloud Computing and Grid Computing 360-Degree Compared,” Grid Computing Environments Workshop, 2008.
3. N. Chowdhury, and R. Boutaba, “A survey of network virtualization,” Computer Networks, vol. 54, pp. 862-876, 2010.
4. B. Mukherjee, Optical WDM Networks. Springer, 2006.
5. N. M. Mosharaf, M. R. Rahman, and R. Boutaba, “Virtual Network Embedding with Coordinated Node and Link Embedding,” IEEE INFOCOM, 2009.
6. M. Rahman, I. Aib, and R. Boutaba, “Survivable Virtual Network Embedding,” NETWORKING, vol. 6091 LNCS, pp. 40-52, 2010.
7. J. Luand, and J. Turner, “Efficient embedding of virtual networks onto a shared substrate,” Washington University, Tech. Report, WUCSE-2006-35, 2006.
8. Y. Zhu, and M. Ammar, “Algorithms for assigning substrate network resources to virtual network components,” in Proceedings of IEEE INFOCOM, 2006.
9. I. Houidi, W. Louati, D. Zeghlache, P. Papadimitriou, and L.

Mathy, "Adaptive virtual network provisioning," in SIGCOMM, New Delhi, pp. 41-48, 2010.

10. W. L. Yeow, C. Westphal, and U. Kozat, "Designing and embedding reliable virtual infrastructures," in Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures, pp. 33-40, 2010.
11. A. Razzaq, and M. S. Rathore, "An Approach Towards Resource Efficient Virtual Network Embedding," 2010 2nd International Conference on Evolving Internet pp. 68-73, 2010.
12. F. E. Zaheer, J. Xiao, and R. Boutaba, "Multi-provider service negotiation and contracting in network virtualization," in Network Operations and Management Symposium (NOMS), pp. 471-478, 2010.
13. S. Zhang, and X. Qiu, "A Novel Virtual Network Mapping Algorithm for Cost Minimizing," Journal of Selected Areas in Telecommunications, pp. 1-9, 2011.
14. H. Yu, V. Anand, C. Qiao, H. Di, and J. Wang, "On the Survivable Virtual Infrastructure Embedding Problem," IEEE ICCCN, 2010.
15. M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: Substrate support for path splitting and migration," ACMSIGCOMM Computer Communication Review, vol. 38, no. 2, pp. 17–29, 2008.
16. H. Yu, C. Qiao, V. Anand, X. Liu, H. Di, and G. Sun, "Survivable Virtual Infrastructure Embedding in a Federated Computing and Networking System under Single Regional failures," IEEE Globecom, 2010.
17. M. Pióro, and D. Medhi, Routing, Flow, and Capacity Design in Communication and Computer Networks. Elsevier, 2004.
18. A. Sen, B. H. Shen, L. Zhou, and B. Hao, "Fault-Tolerance in Sensor Networks: A New Evaluation Metric," IEEE INFOCOM 2006.
19. D. P. Palomar, and M. Chiang, "A tutorial on decomposition methods for network utility maximization," J. Selected Areas in Communications, IEEE Journal, Vol. 24, No. 8, pp. 1439-1451, 2006.

AUTHORS

Gang Sun received his M.En. degree in Signal and Information Processing in 2009 from Chengdu University of Technology. Currently, Gang Sun is pursuing his Ph.D. degree in Communication and Information System at University of Electronic Science and Technology of China. His research interests include network survivability and next generation network.

E-mail: gangsun@uestc.edu.cn

Hongfang Yu received her B.S. degree in Electrical Engineering in 1996 from Xidian University, her M.S. degree and Ph.D. degree in Communication and Information Engineering in 1999 and 2006 from University of Electronic Science and Technology of China, respectively. From 2009 to 2010, she was a Visiting Scholar t the Department of Computer Science and Engineering, University at Buffalo (SUNY). Her research interests include network survivability and next generation Internet, cloud computing etc.

E-mail: yuhf2004@gmail.com

Lemin Li graduated from Jiaotong University, Shanghai, China in 1952, majoring in electrical engineering. From 1952 to 1956 he was with the Department of Electrical Communications at Jiaotong University. Since 1956 he has been with Chengdu Institute of Radio Engineering (now the University of Electronic Science and Technology of China). From August 1980 to Aug. 1982, he was a Visiting Scholar in the Dept. of Electrical Engineering and Computer Science at the University of California at San Diego, USA, doing research on digital and spread spectrum communications. His present research work is in the area of communication networks including broadband networks and wireless networks.

E-mail: lml@uestc.edu.cn

Vishal Anand is an associate professor at The College at Brockport, SUNY. He received his B.S. degree in Computer Science and Engineering from the University of Madras, Madras (Chennai), India in 1996, and the M.S. and Ph.D. degrees in Computer Science and Engineering from the University at Buffalo, SUNY in 1999 and 2003. He has worked as a research scientist at Bell Labs, Lucent technologies and Telcordia Technologies (ex-Bellcore), where he investigated issues relating to traffic routing and survivability in optical networks. He is the recipient of the "Rising Star" and the "Promising Inventor Award" award from the Research Foundation of The State University of New York (SUNY), and the recipient of the "Visionary Innovator" award from the University of Buffalo (SUNY). His research interests are in the area of wired and wireless computer communication networks and protocols, cloud and grid computing.

E-mail: vanand@brockport.edu

Hao Di is a Ph.D. candidate and a research group member in University of Electronic Science and Technology of China. His research interests include next generation network.

E-mail: dihao@uestc.edu.cn

DOI: 10.4103/0256-4602.85968 ; Paper No. TR 89_11;