On the Impact of Resilience, Delay and Geographical Location constraints in the Virtual Network Embedding Problem

Bráulio Antônio Mesquita Souza
Departamento de Ciência da Computação – Universidade Federal de Minas Gerais (UFMG)
Av. Pres. Antônio Carlos, 6627 - Pampulha, Belo Horizonte - MG
`brauliomesquita@dcc.ufmg.br`

Geraldo Robson Mateus
Departamento de Ciência da Computação – Universidade Federal de Minas Gerais (UFMG)
Av. Pres. Antônio Carlos, 6627 - Pampulha, Belo Horizonte - MG
`mateus@dcc.ufmg.br`

Fernanda Sumika Hojo de Souza
Departamento de Ciência da Computação – Universidade Federal de São João del-Rei (UFSJ)
Av. Visconde do Rio Preto, s/n^{o}, Colônia do Bengo - São João del-Rei – MG
`fsumika@ufsj.edu.br`

ABSTRACT

Network virtualization has been applied as an alternative way to deal with a phenomenon known as the "ossification of the Internet", allowing new features to be implemented and tested. Through virtualization, a logical view of the hardware is provided such that multiple virtual networks can operate simultaneously on the same physical resources. In this context, the problem of efficient allocation of resources emerges known as the virtual network embedding problem. Additional constraints may be appended to the original problem in order to be closer to real applications and also provide some quality of service guarantees. In this work, we evaluate the impact of additional constraints under different objectives for the problem through an Integer Linear Programming model. Optimal solutions along with feasible upper bounds demonstrate the constraints affect the acceptance ratio and computational times while different objectives influence the usage of physical resource components.

KEYWORDS. Virtual Network Embedding; Integer Linear Programming; Resilience; Quality of Service.

PAPER TOPICS: Combinatorial Optimization

1. Introduction

The importance of communication networks has grown dramatically over the recent years, as well as greater accessibility to Internet. In general, we are facing evolving technologies that offer a variety of services such as audio transmission, video and data, in addition to remote processing and storage, among others. Moreover, the increasing demand from various network users (people, organizations, institutions, etc) and the concern with service reliability in order to avoid the dissatisfaction of those customers require degrees of organization in the network. Providers of network infrastructure and services then have begun to offer high level services taking into account multi-attribute, including demand capacity, delay, survivability, costs, among others. Those attributes need to be met according to agreements established with network users. Therefore, all those attributes must be processed with more accuracy.

Internet is the biggest communication network aggregation of global scale. It was proposed based on the TCP/IP model whose simplicity has enabled rapid growth and great evolution of applications. However, this simplicity is responsible for limitations that become increasingly evident and compromise its expansion and the inclusion of new features that had not been foreseen in the original project. Some barriers are due to the current IPv4 use, high costs for expansion, difficulty for advancing the development of the "Internet of Things", among others. For example, it is often impossible for hosts to describe failures and their causes. Limitations of the current Internet architecture can been found at [Papadimitriou et al., 2010]. This phenomenon is known as "ossification of the internet", representing a problem for the Future Internet (FI), where alternatives and more flexible architectures are critical in order to create a more efficient and secure network.

Currently, network virtualization has been used as an alternative to circumvent the "ossification", allowing the implementation and run of new features [Fischer et al., 2013]. Although not very new, virtualization is a promising technique to overcome the resistance of the current internet to fundamental changes. Through virtualization, it is possible to create a logical view of hardware components such that multiple virtual networks can operate simultaneously sharing physical resources [Chowdhury e Boutaba, 2010]. During the last decade, virtualization technology has been adopted increasingly in order to improve efficiency and agility of processing and storage over shared resources. Some of the main advantages of virtualization include flexibility, scalability, isolation, cost reduction and safety. In this context, the problem of efficient infrastructure resource allocation to meet virtual requests arises. This problem is known as Virtual Network Embedding (VNE) [Zhu e Ammar, 2006].

The basic version of VNE problem considers a physical network or substrate, composed by physical nodes and links with processing (CPU) and transmission bandwidth capacities, respectively; along with virtual network requests (VNs) demanding CPU and bandwidth resources for virtual nodes and virtual links, respectively. Each virtual node of a VN must be assigned on a single node of the physical substrate, while a physical node cannot have more than one node of the same VN assigned to it. Since objective functions generally minimize bandwidth usage, allowing a physical node to host more than one node from the same virtual request, would tend the virtual nodes to be mapped into the same physical node and therefore no physical links would be used to meet the communication between routers. This constraint is present in other works in literature, e.g. in [Chowdhury e Boutaba, 2010]. On the other hand, a virtual link may be assigned to more than one physical link, i.e., it can be assigned to a physical path as long as all links belonging to the path have enough resources. VNE admits several variants, depending on which constraints are considered. In this work, we present additional constraints that can be tackled to make the problem more real.

Figure 1 shows two virtual networks (VN^1 and VN^2) assigned to the same physical substrate. The network substrate is formed by four nodes and four links with associated capacities respectively. Virtual request VN^1 is composed by three nodes and three links while VN^2 is given by two nodes and one link. The substrate capacities and the virtual network requests are represented in the figure as numbers on the nodes and links, respectively. As one can see, both networks could be embedded on the same substrate respecting all capacities.

Network survivability is a fundamental issue that cannot be disregarded in virtualization scenarios. Under failure conditions in the physical network, protection and regeneration mechanisms allow the network to maintain its maximum connectivity and Quality of Service (QoS). Resilience can reside on the topological level, on protocol design or even through additional resource allocation. In the topological level, for example, a biconnected network is robust against random failure in a single link. Other examples include the use of network protocols to reallocate requests dynamically when a failure occurs. Protection mechanisms proactively allocate extra resources to avoid loss of service in failure conditions.

Virtualization presents several challenges to overcome, bringing out problems as VNE, which is known to be NP-hard, reducible to the multi-way separator problem [Andersen, 2002]. Besides, virtualization needs to be as realistic as possible, incorporating into models and algorithms, attributes and assumptions closer to the real problem and scenarios in which failures may happen.

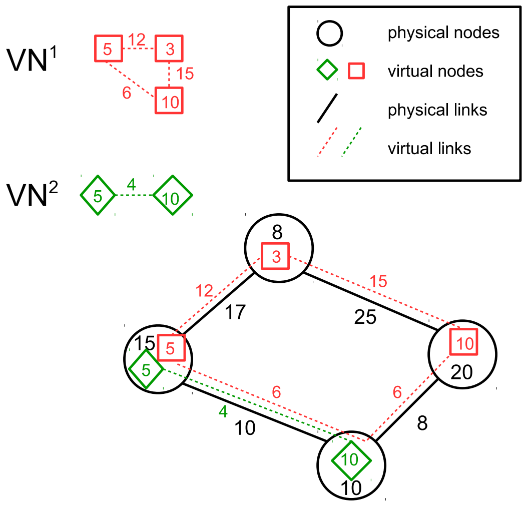

Figure 1: Two VNs embedded in the substrate

Optimization techniques [Magnanti e Wong, 1984], [Resende e Pardalos, 2005], [Chinneck et al., 2009] play a key role on this task, seeking for the improvement of efficiency, scalability and security of these networks, and also meeting the demands of users at lower costs.

This work presents an Integer Linear Programming model for the Virtual Network Embedding problem with Resilience, Delay and Geographical Location constraints under three main objectives: minimization of bandwidth allocation, maximization of load balance and minimization of delay. Although these constraints have already been presented in literature, they haven't been evaluated individually and in their combinations as presented in this paper. Computational results demonstrate the additional constraints affect the acceptance ratio and computational times while different objectives influence the usage of physical resource components. Section 2 describes related works in the area, while Section 3 formally defines the problem. Section 4 presents the proposed models and Section 5 shows computational results and discussion. Finally, Section 6 gives the conclusion and future work.

2. Related Work

Concerned about the Future Internet, much research have been developed over the VNE problem [Zhu e Ammar, 2006], [Yu et al., 2008], [Inführ e Raidl, 2011], [Chowdhury et al., 2009], [Cheng et al., 2011]. According to [Fischer et al., 2013], the VNE can be classified according to reconfiguration of VNs (static vs. dynamic), algorithm strategy (centralized vs. distributed) and fault tolerance (concise vs. redundant). This taxonomy categorizes the several works from literature.

The dynamic and static models allow VNs to be reallocated or not, respectively, i.e., once allocated they can have their position changed or they are kept on the same place until the end of their lifetime. In dynamic models, the cost of reconfiguration of VNs needs to be considered, but with the benefit to improve the fragmentation of the substrate [Fajjari et al., 2011], [Sun et al., 2013].

The embedding algorithm can operate through a centralized or distributed strategy. While a single entity is responsible for the assignment task in the centralized model, many entities of the network can share the assignment task in a distributive manner. The global overview provided by the centralized model allows better solutions to be generated, at a higher processing cost. On the other hand, a distributed model has an overhead for exchanging messages among components, but also provides scalability in large scale scenarios.

Failures occur in practice, requiring networks to provide resilience when a substrate node or link fails during operation [Rahman e Boutaba, 2013; Oliveira et al., 2015]. In a concise model, the resources allocated from the substrate are exactly those required by VNs, i.e., no additional resource is reserved. Redundant models can reserve extra resources to deal with faults, which usually increases the rejection of new requests depending on the resource reservation. A binary quadratic programming formulation and two heuristic algorithms that consider protection against node failure are presented in [Guo et al., 2014].

The virtual network embedding problem has an online nature, in which requests arrive at the service provider dynamically, and it is not possible to know all demands *a priori*. However, it can be solved by waiting for a set of VNs and treating then all together in the algorithm. It is important to point out that even the *offline* approach is NP-hard.

In [Chowdhury et al., 2012], authors solve the *online* version of VNE using a time window, i.e. the VNs are queued and then processed in batches instead of being mapped as they arrive. As several VNs must be mapped onto the substrate network at the same time and the current available resources might not be enough to host them, some VNs must be rejected.

Heuristics for VNE are proposed in [Zhu e Ammar, 2006], aiming to provide load balance and minimize node and link usage. Authors compared approaches with reconfiguration and without reconfiguration, showing that the proposed algorithm outperforms the *least load* algorithm from literature and allowing reconfiguration for a small subset of critical VNs reach most benefits of dynamic reconfiguration at a low cost.

In [Chowdhury et al., 2009], two algorithms are proposed: D-ViNE (*Deterministic VN Embedding*) and R-ViNE (*Randomized VN Embedding*). These algorithms are based on a two stage procedure in which the problem is solved through linear programming disregarding integer constraints (VNE_LP_RELAX), followed by a deterministic or a random rounding stage for node mapping, respectively. Link mapping is performed in the sequel by one of the two procedures: shortest path when a single path is considered for a virtual link or multicommodity flows, when splitting in more than one physical path is allowed for a virtual link. Computational results show that flow splitting increases the acceptance rate of requests because bottlenecks can be avoided; however, most real applications do not allow this flexibility in practice.

Additional constraints may be considered in VNE in order to make the problem closer to a real application. In [Inführ e Raidl, 2011], authors take into account the geographical location of nodes, a maximum delay allowed in the communication of two virtual nodes and routing capacity in physical nodes. Location constraints are incorporated by [Hu et al., 2012] into an ILP model to solve small instances of the survivable network embedding problem, whilst an efficient heuristic algorithm is proposed to solve greater instances. An evaluation of the impact of location constraints with different substrate network topologies is presented in [Luizelli et al., 2013]. For instance, geographical location constraints can guarantee that an online game service owns virtual nodes in strategic cities as well as delay constraints ensure quality of service in terms of maximum delay. Routing constraints limit the routing capacity of nodes, as they are not able to route the full bandwidth they are connected to in practice.

Most works in the literature consider that infrastructure providers are operating all time. In [Rahman e Boutaba, 2013], authors study resilience and network survivability, when links fail in the physical substrate. Authors propose a mixed integer program formulation along with efficient heuristics for the survivable virtual network embedding problem. The main goal regards the link mapping phase, while for the node mapping phase heuristics from literature are employed. Heuristics work in a proactive strategy, in which extra resources are allocated as a protection mechanism to avoid failures in applications where delay is critical. On the other hand, a reactive strategy, which avoid wasting additional resources reservation by treating the fail after its occurrence, leads to a short delay in the survivability mechanism.

The approach considered in this work is classified as static for the reconfiguration of VNs, centralized for the algorithm strategy, concise and redundant for fault tolerance (both cases are evaluated), *offline* to the knowledge of VNs, allows the mapping of a subset of VNs and does not split a virtual link into multiple physical paths. To the best of our knowledge, the evaluation of the impact induced by different combinations of appended constraints under different objective functions is not provided by the works in literature.

3. Problem Definition

The Virtual Network Embedding problem can be modeled through Integer Linear Programming (ILP) as follows. Let $G^s = (N^s, E^s)$ be a weighted undirected graph representing the substrate, in which N^s and E^s consist of the set of physical nodes and links, respectively. Each substrate node $i \in N^s$ has a CPU capacity C_i and each substrate link $(i, j) \in E^s$ has a bandwidth capacity B_{ij}. Similarly, a virtual network request $v \in V$ is modeled as a undirected graph $G^v = (N^v, E^v)$, composed by a set of virtual nodes (N^v) and links (E^v) associated to CPU (c^v_k) and bandwidth (b^v_{kl}) demands, respectively.

A VNE solution consists in finding an assignment $f : G^v \rightarrow G^s$, in which $\forall\ k \in N^v\ \exists\ i \in N^s : c^v_k \leq C_i$ and $\forall\ (k, l) \in\ E^v\ \exists\ \{(i, j)^1, ..., (i, j)^m\} : b^v_{kl} \leq B^1_{ij}, ..., B^m_{ij})$. If such an embedding is possible, the request is said to be accepted; otherwise, it is refused.

In order to provide resilience against single link failure, two edge disjoint paths are necessary to connect each virtual link of the request. Therefore, in the redundant strategy, each virtual link must be assigned to two edge disjoint paths in the physical substrate.

4. ILP Model

In order to model the Virtual Network Embedding problem by ILP, let the decision variables be:

- $y^v \in \{0, 1\}$ - indicates whether virtual network request $v \in V$ is accepted ($y^v = 1$) or not ($y^v = 0$).
- $z^v_{ki} \in \{0, 1\}$ - indicates whether virtual node $k \in N^v$ from virtual request $v \in V$ is assigned to substrate node $i \in N^s$ assuming value 1 and 0 otherwise.
- $x^{vkl}_{ij} \in \{0, 1\}$ - indicates whether virtual link $(k, l) \in E^v$ from virtual request $v \in V$ traverses substrate link $(i, j) \in E^s$ along the path connecting substrate nodes assigned to $k, l \in N^v$.

Let M be a penalty for not attending a request.

Considering the minimization of total bandwidth (VNE_BW) consumed from the substrate as the objetive function, the problem can be stated as follows:

VNE_BW:

$$min \left[\sum_{v \in V} \sum_{(k,l) \in E^v} \sum_{(i,j) \in E^s} b^v_{kl} x^{vkl}_{ij} + M \times \sum_{v \in V} (1 - y^v) \right] \tag{1}$$

Subject to the following sets of constraints:

$$\sum_{i \in N^s} z^v_{ki} \geq y^v \qquad \forall k \in N^v, \forall v \in V \tag{2}$$

$$\sum_{k \in N^v} z^v_{ki} \leq 1 \qquad \forall i \in N^s, \forall v \in V \tag{3}$$

$$\sum_{v \in V} \sum_{k \in N^v} c^v_k z^v_{ki} \leq C_i \qquad \forall i \in N^s \tag{4}$$

$$\sum_{v \in V} \sum_{(k,l) \in E^v} b^v_{kl} x^{vkl}_{ij} \leq B_{ij} \qquad \forall (i, j) \in E^s \tag{5}$$

$$\sum_{(i,j)\in E^s} x_{ij}^{vkl} - \sum_{(h,i)\in E^s} x_{hi}^{vkl} = z_{ki}^v - z_{li}^v \qquad \forall i \in N^s, \forall (k,l) \in E^v, \forall v \in V \tag{6}$$

$$y^v \in \{0,1\} \qquad \forall v \in V \tag{7}$$

$$z_{ki}^v \in \{0,1\} \qquad \forall i \in N^s, \forall k \in N^v, \forall v \in V \tag{8}$$

$$x_{ij}^{vkl} \in \{0,1\} \qquad \forall (i,j) \in E^s, \forall (k,l) \in E^v, \forall v \in V \tag{9}$$

Objective function (1) minimizes the total amount of bandwidth allocated for the virtual network requests in the substrate network. Moreover, if a virtual request is refused, a penalty M is added to the objective function. Note that the total CPU is always constant for a VN, while the bandwidth depends on how many physical links are used by the assignment of a VN. Therefore, only the bandwidth is taken into account in the objective.

Constraints (2) guarantee all virtual nodes of an accepted network request to be mapped into a node in the substrate network. Constraints (3) ensure a substrate node is assigned only a single virtual node from each network request. Constraints to guarantee that the CPU capacity of the substrate nodes will not be exceeded by the CPU requirements from the virtual networks are given in (4). Similarly, constraints (5) ensure the bandwidth capacity for the substrate links. Constraints (6) guarantee the connectivity between the virtual nodes in the substrate network through the mapping of virtual links using single substrate paths. Contraints (7), (8) and (9) indicate the domain of the decision variables y, z and x, respectively.

In order to analyze different solutions for VNE, we propose two alternative objective functions. The first one (VNE_LB) provides load balance regarding bandwidth allocation in the network and can be given by:

VNE_LB:

$$min \left[\alpha + M \times \sum_{v\in V} (1 - y^v) \right] \tag{10}$$

It also requires an additional set of constraints:

$$\alpha \geq \frac{\sum_{v\in V} \sum_{(k,l)\in E^v} b_{kl}^v x_{ij}^{vkl}}{B_{ij}} \qquad \forall (i,j) \in E^s \tag{11}$$

Let α be the maximum link utilization among all physical links. Objective function (10) along with constraints (11) minimizes the maximum link utilization of all links, which tends to distribute the requests over the substrate network, avoiding overloading some substrate links. The penalty is also applied every time a request is refused.

The second alternative objective function (VNE_DL) aims to minimize the delay in the communication of all components of virtual requests. Consider a parameter d_{ij} for all $(i,j) \in E^s$ as the delay observed in such a substrate link.

For that, the objective can be expressed as:

VNE_DL:

$$min \left[\sum_{v\in V} \sum_{(k,l)\in E^v} \sum_{(i,j)\in E^s} d_{ij} x_{ij}^{vkl} + M \times \sum_{v\in V} (1 - y^v) \right] \tag{12}$$

Objective function (12) minimizes the total delay over all virtual requests. This objective tends to assign virtual nodes of the same request closer to reduce the delay among them. The penalty is also applied every time a request is refused.

Aiming to create a scenario closer to real applications, additional constraints may be used in order to provide some QoS guarantees. In the following, we present three sets of constraints that can be appended to both models in an attempt to capture more particularities of the virtualization context.

1. **Resilience constraints (R)**

   Failures are common in network scenarios and resilience is a fundamental parameter concerning QoS. Former studies showed that single link failures represent 70% of failures in networks [Iannaccone et al., 2002]. Although node failures are also important, before addressing them, a solution for link failures has to be already figured out, since a node failure solution depends on tolerating adjacent link failures. In order to ensure resilience against single link failure in the substrate network, two link disjoint substrate paths must exist for each virtual network request. Since variables x_{ij}^{vkl} are binary, multiplying the right-hand side of constraint (6) guarantees two edge-disjoint physical paths for each virtual link. Thus, constraints (6) would read as:

   $\forall i \in N^s, \forall (k,l) \in E^v, \forall v \in V$:

$$\sum_{(i,j)\in E^s} x_{ij}^{vkl} - \sum_{(h,i)\in E^s} x_{hi}^{vkl} = 2(z_{ki}^v - z_{li}^v) \qquad (13)$$

2. **Geographical Location constraints (L)**

   Geographical location constraints can be specially important depending on the user requirements. For example, when security and confidentiality are priority, the user may not desire that his data would be hosted to specific physical nodes. On the other hand, a user may want to have its application more spread, increasing availability due to geographical diversity. To figure out that problem, service providers can include this additional constraint as a parameter of the service level agreement made with their customers, allowing them to decide the locations where their data will be hosted.

   For that, consider the geographical location of physical nodes and the preferential geographical location of virtual nodes. Assume R^v as the maximum radius a virtual node can be far from its preferential location and d_{ki}^v as the euclidean distance from a physical node $i \in N^v$ to the preferential location of virtual node $k \in N^s$ from the network $v \in V$. Thus, constraints (14) will ensure that a virtual node is assigned only to a physical node belonging to the set $I^s \subseteq N^s$ such that $i \in I^s$ iff $d_{ki}^v \leq R^v$.

   $\forall k \in N^v, \forall v \in V$:

$$\sum_{i\in I^s} z_{ki}^v \geq y^v \qquad (14)$$

3. **Delay constraints (D)**

   Delay constraints are directly related to the communication quality expected between a pair of nodes. For some critical applications, this parameter should have a higher priority. Therefore, consider parameters d_{ij} as the delay observed in a substrate link $(i,j) \in E^s$ and D_{kl}^v as the maximum delay admitted by virtual link $(k,l) \in N^v$. Constraints (15) ensure the sum of delays of all substrate links which the virtual link traverses will not exceed D_{kl}^v.

   $\forall (k,l) \in E^v, \forall v \in V$:

$$\sum_{(i,j)\in E^s} d_{ij} x_{ij}^{vkl} \leq D_{kl}^v \qquad (15)$$

It is important to mention that when Delay is considered along with Resilience constraints, the right-hand side of equation (15) must be multiplied by 2. Hence, the delay constraint imposes an average delay limit for the paths that meet a request.

5. Computational Results

In order to evaluate the performance of the proposed models along with additional constraints, they were implemented using CPLEX release 12.6.1 [ILOG]. Instances[1] for the VNE used during the tests were proposed on [Chowdhury et al., 2009]. The substrate network used is composed by 50 vertices positioned in a 25×25 grid connected with a probability of 0.5, randomly generated by GT-ITM tool. CPU and bandwidth resources correspond to real numbers following an uniform distribution between 50 and 100. Four instances were created selecting different sizes for the set of VN requests: 5, 10, 15 and 20. The number of nodes in each VN is randomly determined between 2 and 10, by an uniform distribution. The average connectivity rate of a VN is fixed at 50%. CPU requirements of virtual nodes are uniformly distributed between 0 and 20 and bandwidth requirements of virtual links between 0 e 50. Virtual nodes are also located in a 25×25 grid.

Tests were run for objectives VNE_BW, VNE_LB and VNE_DL, along with the additional constraints mentioned in Section 4, which were added separately and in combination. Tables 1, 2 and 3 summarize results for all instances and the three objectives, respectively. The first column indicates the instance according to the size of the set of requests $|V| \in \{5, 10, 15, 20\}$. The second column shows the additional constraints considered: Resilience (R), Geographical Location (L) and Delay (D), and their combinations. The 'Objective' column refers to the solution cost according to the objective function used. It is important to note that penalties are not considered in values shown in the tables. The 'Acc. Ratio $(\%)$' column indicates the percentage of virtual networks whose demands were met. The next two columns show the percentage of links used in the solution and average usage ratio of these links. The same columns are presented regarding the node usage. Finally, 'Time (sec.)' presents a value in seconds of the time spent. Values 3600 and over indicate the algorithm was stopped because the time limit of 1 hour run was reached.

Table 1 shows that larger instances (increasing the number of requests) present a higher objective function considering cases that all requests are accepted. Also, it can be observed that the acceptance ratio tends to decrease when the instance size increases, for all cases. Considering the original model, i.e., with no additional constraints, all instances presented were able to have their requests completely embedded by the algorithm. However, the addition of other constraints makes the acceptance ratio to decrease. It can be seen that when constrains are added individually, the acceptance ratio remains the same, except for the instance with 20 VN requests. When constraints are added in combination with others, the acceptance ratio decreases significantly, indicating that it is hard to meet all constraints simultaneously. Note that the combination "L+D+R" leads to the smaller acceptance ratio in all instances. The execution time shows the problem can be solved to optimality in a few seconds when the Location constraint is considered. This can be explained due to the smaller search space of feasible solutions when nodes are restricted to specific geographical locations, decreasing the number of possibilities. When the Location constraint is not considered, the algorithm reached the time limit imposed, returning the best feasible solution found so far. Considering the percentage of used nodes, it can be noted that values are similar for all variations in each instance, except in cases that the acceptance ratio is smaller, leading to fewer used nodes. The average node usage is similar independently. For the percentage of used links and average link usage, it is important to point out that all versions including resilience present higher values, according to the acceptance ratio. As providing survivability consumes more resources to allocate extra bandwidth, the link usage is higher in those cases.

According to Table 2, the problem becomes more difficult when load balance is considered in the objective function. It can be seen that the acceptance ratio decreases in most cases

[1]http://www.mosharaf.com/ViNE-Yard.tar.gz

Table 1: Minimize Bandwidth (VNE_BW)

# VNs	Con.	Objective	Acc. Ratio	Used Links (%)	Avg. Link Usage (%)	Used Nodes (%)	Avg. Node Usage (%)	Time (sec.)
5	D	741.23	100.00	12.20	33.94	34.00	15.53	3602.37
	D R	2269.56	100.00	26.02	44.44	34.00	16.72	3602.85
	R	2331.74	100.00	30.08	42.51	40.00	13.74	3603.42
	L	1066.98	100.00	17.07	34.82	38.00	14.23	0.19
	L D	492.50	80.00	8.94	34.88	22.00	14.19	0.25
	L D R	803.30	60.00	13.01	35.18	14.00	15.24	0.57
	L R	3307.31	100.00	39.43	45.33	34.00	16.76	5.62
	-	741.23	100.00	12.20	32.14	34.00	17.59	3602.10
10	D	1346.99	100.00	21.14	34.25	50.00	23.71	3605.03
	D R	1770.50	80.00	19.51	46.88	30.00	25.18	3603.11
	R	4784.90	100.00	49.59	50.89	52.00	23.98	3600.10
	L	1667.13	100.00	21.14	42.62	46.00	24.81	0.44
	L D	1089.40	90.00	16.67	37.32	38.00	21.65	0.77
	L D R	1653.10	60.00	17.89	50.75	16.00	35.31	4.92
	L R	5821.60	100.00	62.20	51.31	50.00	24.02	584.23
	-	1346.99	100.00	20.33	38.76	54.00	21.62	3600.03
15	D	1683.84	100.00	25.20	39.42	60.00	21.56	3602.75
	D R	4252.55	86.67	40.24	54.23	44.00	23.55	3602.36
	R	5720.89	100.00	56.10	54.04	62.00	21.30	3600.19
	L	2155.55	100.00	26.02	47.85	52.00	24.62	0.67
	L D	722.35	53.33	8.13	52.38	26.00	20.25	0.72
	L D R	856.00	20.00	8.94	54.71	12.00	15.02	1.43
	L R	8013.70	100.00	70.33	62.16	58.00	22.20	3600.05
	-	1683.84	100.00	26.02	35.64	60.00	22.03	3600.92
20	D	2635.32	100.00	36.59	38.40	64.00	30.93	3600.04
	D R	2056.00	25.00	23.58	46.89	26.00	19.97	3600.10
	R	7876.20	80.00	71.95	57.80	60.00	24.39	3600.06
	L	3642.85	100.00	33.33	59.97	50.00	37.91	11.72
	L D	2077.20	70.00	21.14	54.70	46.00	27.02	1.89
	L D R	2164.00	25.00	21.95	53.28	16.00	25.58	31.54
	L R	7654.60	80.00	72.36	56.95	52.00	25.65	3600.03
	-	2635.32	100.00	34.15	43.79	80.00	23.27	3600.00

compared to VNE_BW solutions. In general, the number of links and nodes is superior to the ratios of solutions provided by VNE_BW, however the average link and node usage of VNE_LB solutions are lower, showing that virtual links should be spread in the physical network. Regarding the execution time, the number of cases solved to proven optimality is bigger than VNE_BW. It is possible to note that cases which reached the time limit involve the resilience constraints.

The summarized results for VNE_DL are shown in Table 3. It is important to note that the addition of the Delay constraint affects the solution even when the objective function aims a minimum delay. The objective function VNE_DL minimizes the total delay of all VN requests, while the Delay constraint imposes delay limits to individual paths. VNE_BW and VNE_DL have in most cases, similar acceptance ratios, although VNE_DL achieves higher acceptance ratios for few cases with the instance of 20 VNs. Another interesting observation concerns the components usage. The percentage of nodes and links used is lower than VNE_BW and VNE_LB, while the average usage is higher than VNE_BW and VNE_LB. This behaviour indicates that VN components are assigned close to each other overloading a particular set of physical nodes and links to meet the delay minimization. Again, it is possible to note that the number of cases solved to proven optimality is bigger than VNE_BW and cases which reached the time limit involve the resilience constraints.

6. Conclusion and Future Work

The impact of additional constraints under different objectives was evaluated for the Virtual Network Embedding problem through an Integer Linear Programming model. Resilience, geographical location and delay constraints were tested along with three objectives. VNE_BW minimizes the total bandwidth allocation, VNE_LB minimizes the maximum link utilization, producing load balance, while VNE_DL minimizes the delay in the communication of all requests. Additional constraints

Table 2: Load Balance (VNE_LB)

# VNs	Con.	Objective	Acc. Ratio	Used Links (%)	Avg. Link Usage (%)	Used Nodes (%)	Avg. Node Usage (%)	Time (sec.)
5	D	0.46	100.00	19.92	27.99	40.00	13.61	18.19
	D R	0.73	100.00	42.68	39.92	42.00	13.54	3600.06
	R	0.55	100.00	87.40	35.23	38.00	13.93	3600.04
	L	0.52	100.00	49.19	28.81	36.00	14.92	8.56
	L D	0.60	80.00	11.38	25.20	20.00	14.01	0.40
	L D R	0.70	60.00	21.14	32.64	14.00	17.00	0.67
	L R	0.77	100.00	93.09	53.60	32.00	17.73	56.58
	-	0.46	100.00	30.49	25.92	34.00	16.06	44.51
10	D	0.49	100.00	30.89	28.67	60.00	19.46	76.88
	D R	0.60	60.00	19.92	26.42	26.00	22.47	3600.05
	R	0.88	100.00	94.31	53.95	60.00	19.75	3600.06
	L	0.52	100.00	74.80	31.87	54.00	21.96	33.83
	L D	0.80	90.00	21.14	29.72	40.00	20.74	0.81
	L D R	0.80	60.00	44.31	35.10	20.00	27.14	188.82
	L R	0.97	100.00	94.72	58.59	54.00	21.43	3337.09
	-	0.48	100.00	55.28	24.40	60.00	19.59	553.60
15	D	0.49	100.00	34.15	30.82	68.00	18.99	8.31
	D R	0.95	33.33	12.20	45.17	18.00	13.72	3600.06
	R	0.95	93.33	93.09	60.81	70.00	17.10	3600.04
	L	0.55	100.00	86.18	38.07	58.00	21.84	434.53
	L D	0.75	53.33	10.16	46.70	24.00	22.33	2.04
	L D R	1.00	20.00	47.97	42.20	12.00	15.82	36.52
	L R	0.95	93.33	95.53	63.46	58.00	22.51	3600.06
	-	0.48	100.00	71.14	30.93	72.00	18.54	240.10
20	D	0.49	100.00	54.88	31.37	70.00	27.69	941.53
	D R	1.00	20.00	40.00	31.01	22.00	14.02	3599.0
	R	1.00	20.00	36.59	33.53	18.00	16.65	3601.58
	L	0.68	100.00	82.11	42.20	66.00	29.51	3600.05
	L D	1.00	70.00	29.67	51.31	42.00	29.09	3.93
	L D R	1.00	25.00	72.76	52.17	18.00	24.34	2324.46
	L R	1.00	80.00	95.53	60.31	46.00	29.50	3600.08
	-	0.68	100.00	90.24	42.05	78.00	25.19	3600.06

were appended to the model individually but also in combination, showing the characterization on how solutions can be affected in terms of acceptance ratio, usage of physical components and computational time.

Computational experiments showed that location constraints make the problem easier to be solved by restricting the possibilities of node assignment. As we allow assigning virtual nodes in a higher number of substrate nodes, the acceptance ratio of request increases, and also the problem difficulty. When resilience against single link failure is considered, the problem becomes more complex, with lower acceptance ratio because of the higher use of the substrate resources. Delay constraints did not show a significant impact. Moreover, when constraints are added in combination with others, the acceptance ratio usually decreases, indicating that it is hard to meet all constraints simultaneously. The combination "L+D+R" led to the smaller acceptance ratio in all instances. Regarding the different objectives, the main impact occurs in the usage of substrate resources. A load balance solution spreads the allocations over the whole network leading to a higher number of components in use, but at lower rates of usage. A minimum delay solution concentrates the allocations over closer physical components leading to a higher average usage. Finally, the bandwidth minimization solution seems an intermediate usage between the other ones.

As future work, the online version of the problem can be considered, dealing with virtual requests as they arrive. Moreover, reconfiguration of previously allocated VNs should be tackled to provide less fragmentation in the substrate. The identification of critical components will also allow resizing the network in strategic points aiming to increase the acceptance ratio. Advanced optimization techniques and metaheuristics may also be explored in order to provide scalability in terms of number and size of VN requests, without loss in solution quality.

Table 3: Minimize Delay (VNE_DL)

# VNs	Con.	Objective	Acc. Ratio	Used Links (%)	Avg. Link Usage (%)	Used Nodes (%)	Avg. Node Usage (%)	Time (sec.)
5	D	73.78	100.00	9.76	43.10	30.00	19.30	224.94
	D R	362.07	100.00	24.39	56.29	32.00	17.63	3602.97
	R	367.19	100.00	22.76	59.25	30.00	17.42	3604.20
	L	368.33	100.00	17.89	36.92	32.00	17.14	0.17
	L D	182.20	80.00	8.54	34.94	20.00	14.01	0.23
	L D R	194.00	60.00	15.85	30.15	14.00	16.81	0.84
	L R	1007.21	100.00	39.02	50.55	30.00	18.13	6.87
	-	73.78	100.00	9.76	42.25	30.00	19.32	113.89
10	D	127.56	100.00	13.82	50.92	42.00	29.40	667.69
	D R	295.70	80.00	17.07	65.02	28.00	26.34	3603.39
	R	804.02	100.00	43.50	64.14	54.00	21.72	3600.05
	L	525.21	100.00	21.95	46.74	36.00	32.39	0.55
	L D	347.20	90.00	13.82	48.29	30.00	27.42	0.48
	L D R	398.80	60.00	18.29	50.46	16.00	34.03	2.53
	L R	1619.22	100.00	53.66	63.71	46.00	26.03	460.90
	-	125.49	100.00	13.82	52.69	40.00	31.06	1391.71
15	D	145.47	100.00	12.20	75.66	38.00	37.69	1300.00
	D R	485.85	66.67	28.05	57.13	40.00	20.95	3600.06
	R	1633.12	100.00	67.48	59.60	70.00	20.06	3600.05
	L	861.52	100.00	26.02	54.87	46.00	28.21	0.67
	L D	90.25	53.33	9.76	48.25	26.00	20.58	0.61
	L D R	126.00	20.00	10.57	52.89	10.00	17.78	2.13
	L R	2606.27	100.00	68.29	67.42	54.00	23.72	3600.03
	-	145.47	100.00	13.82	67.87	42.00	33.66	2180.81
20	D	264.14	100.00	20.33	72.02	56.00	36.03	3603.33
	D R	1260.50	85.00	52.44	58.62	50.00	31.59	3636.69
	R	2075.40	95.00	68.70	67.49	66.00	26.99	3659.57
	L	1379.76	100.00	38.21	64.84	52.00	36.14	10.92
	L D	558.40	70.00	20.33	60.79	38.00	32.33	1.20
	L D R	399.00	25.00	21.95	61.63	18.00	24.73	31.74
	L R	2451.00	80.00	62.60	63.34	48.00	28.03	3600.06
	-	273.28	100.00	23.58	61.05	60.00	33.40	3643.93

References

Andersen, D. G. (2002). Theoretical approaches to node assignment. URL `http://www.cs.cmu.edu/~dga/papers/andersen-assign.ps`. Unpublished Manuscript.

Cheng, X., Su, S., Zhang, Z., Wang, H., Yang, F., Luo, Y., e Wang, J. (2011). Virtual network embedding through topology-aware node ranking. *SIGCOMM Comput. Commun. Rev.*, 41(2): 38–47. ISSN 0146-4833.

Chinneck, J. W., Kristjansson, B., e Saltzman, M. J. (2009). *Operations Research and Cyber-Infrastructure*. Springer Publishing Company, Incorporated, 1 edition. ISBN 038788842X, 9780387888422.

Chowdhury, M., Rahman, M. R., e Boutaba, R. (2012). Vineyard: Virtual network embedding algorithms with coordinated node and link mapping. *IEEE/ACM Transactions on Networking*, p. 206–219.

Chowdhury, N. M. M. K., Rahman, M. R., e Boutaba, R. (2009). Virtual network embedding with coordinated node and link mapping. In *INFOCOM*, p. 783–791. IEEE.

Chowdhury, N. M. K. e Boutaba, R. (2010). A survey of network virtualization. *Computer Network*, 54(5):862–876. ISSN 1389-1286.

Fajjari, I., Aitsaadi, N., Pujolle, G., e Zimmermann, H. (2011). VNR algorithm: A greedy approach for virtual networks reconfigurations. In *GLOBECOM*, p. 1–6. IEEE. ISBN 978-1-4244-9266-4.

Fischer, A., Botero, J. F., Beck, M. T., e de Meer, H. (2013). Virtual network embedding: A survey. *IEEE Communications Surveys & Tutorials*, p. 71–97.

Guo, B., Qiao, C., Wang, J., Yu, H., Zuo, Y., Li, J., Chen, Z., e He, Y. (2014). Survivable virtual network design and embedding to survive a facility node failure. *Lightwave Technology, Journal of*, 32(3):483–493.

Hu, Q., Wang, Y., e Cao, X. (2012). Location-constrained survivable network virtualization. In *Sarnoff Symposium (SARNOFF), 2012 35th IEEE*, p. 1–5.

Iannaccone, G., Chuah, C.-n., Mortier, R., Bhattacharyya, S., e Diot, C. (2002). Analysis of link failures in an IP backbone. In *Proceedings of the 2Nd ACM SIGCOMM Workshop on Internet Measurement*, IMW '02, p. 237–242, New York, NY, USA. ACM. ISBN 1-58113-603-X.

ILOG, I. CPLEX Optimizer. URL `http://www-01.ibm.com/software/integration/optimization/cplex-optimizer`.

Inführ, J. e Raidl, G. R. (2011). Introducing the virtual network mapping problem with delay, routing and location constraints. In Pahl, J., Reiners, T., e Voß, S., editors, *INOC*, volume 6701 of *Lecture Notes in Computer Science*, p. 105–117. Springer. ISBN 978-3-642-21526-1.

Luizelli, M. C., Bays, L. R., Buriol, L. S., Barcellos, M. P., e Gaspary, L. P. (2013). Characterizing the impact of network substrate topologies on virtual network embedding. In *Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013)*, p. 42–50.

Magnanti, T. L. e Wong, R. T. (1984). Network design and transportation planning: Models and algorithms. *Transportation Science*, 18:1–56.

Oliveira, R. R., Marcon, D. S., Bays, L. R., Neves, M. C., Gaspary, L. P., Medhi, D., e Barcellos, M. P. (2015). Opportunistic resilience embedding (ore): Toward cost-efficient resilient virtual networks. *Computer Networks*, 89:59–77.

Papadimitriou, D., Tschofenig, H., Rosas, A., Zahariadis, S., et al. (2010). Fundamental limitations of current internet and the path to future internet. *European Commission, FIArch Group, Ver*, 1:80.

Rahman, M. R. e Boutaba, R. (2013). SVNE: Survivable virtual network embedding algorithms for network virtualization. *IEEE Transactions on Network and Service Management*, 10(2):105–118.

Resende, M. G. e Pardalos, P. M., editors (2005). *Handbook of Optimization in Telecommunications*. Springer, New York, NY, USA.

Sun, G., Yu, H., Anand, V., e Li, L. (2013). A cost efficient framework and algorithm for embedding dynamic virtual network requests. *Future Generation Computer Systems*, 29(5):1265–1277. ISSN 0167-739X.

Yu, M., Yi, Y., Rexford, J., e Chiang, M. (2008). Rethinking virtual network embedding: substrate support for path splitting and migration. *Computer Communication Review*, 38(2):17–29.

Zhu, Y. e Ammar, M. (2006). Algorithms for assigning substrate network resources to virtual network components. *INFOCOM 2006. 25th IEEE International Conference on Computer Communications*, p. 1–12.