Stochastic Virtual Network Embedding via Accelerated Benders Decomposition

Behrooz Farkiani[a], Bahador Bakhshi[a,*], S. Ali MirHassani[b]

[a] *Computer Engineering and Information Technology Department, Amirkabir University of Technology, Hafez Avenue, Tehran, Iran*
[b] *Mathematics and Computer Science Department, Amirkabir University of Technology, Hafez Avenue, Tehran, Iran*

Abstract

The interest in network virtualization as a solution to the current ossification of the Internet has been recently revived. Virtual Network Embedding (VNE) is the central problem in network virtualization that determines mapping of Virtual Networks (VN) on the physical substrate network while satisfying VN requirements and the substrate network resource constraints. Much research has been conducted on the VNE problem with the assumption that the VN requirements are known beforehand. Nevertheless, the precise amounts of the requirements are uncertain in practice. The limited research that has considered the uncertainty focused on obtaining *conservative* solutions that increase embedding cost in terms of substrate network resource consumption. In this paper, we examine the VNE problem with the objective of minimizing embedding cost via obtaining a non-conservative solution in the case of the bandwidth requirements of virtual links are expressed as random variables with known distributions. For the first time, we formulate the problem using the stochastic programming framework and utilize the sample average approximation technique to solve it. To tackle its complexity, we decompose the problem based on Benders method which is accelerated by four proposed techniques. Extensive simulation results show that the proposed approach outperforms the conservative solution by about 40-45%, and it is up to 7.5 times faster than the branch and bound method.

Keywords: accelerated Benders decomposition, stochastic programming, uncertainty, virtual network embedding, online algorithms, optimization

1. Introduction

Ever increasing demand for customized services with varying requirements and different level of Quality-of-Service guarantees, leads to escalating demand for networks with customized topologies. In this trend, Internet Service Providers (ISPs) need to change their traditional architecture to satisfy the various requirements. One of the solutions to achieve this goal is virtualization of net-

*Corresponding author
Email addresses: behrooz.farkiani@aut.ac.ir (Behrooz Farkiani), bbakhshi@aut.ac.ir (Bahador Bakhshi), a_mirhassani@aut.ac.ir (S. Ali MirHassani)

work infrastructure that enables ISPs to deploy virtual networks with desired topologies without changes in the underlying network infrastructure. Its advantages are twofold [1]. First, because of isolation of virtualized resources, each customer would be able to administer his/her virtual network independent of other virtual networks. Second, utilization of substrate resources increases as well due to resource sharing.

In this paradigm, ISP's business model is changing, and ISP role would be decoupled into two roles: Service Provider (SP) and Infrastructure Provider (InP). InPs administer the substrate network and put it at the disposal of SPs, on which they could define and deploy Virtual Networks (VNs) and render services to customers thereby [1]. A customer's Virtual Network Request (VNR) is composed of a set of virtual nodes, virtual links, and their required capacities. InP is responsible to map virtual nodes on physical nodes and virtual links to existing paths in the substrate network considering substrate network resource limitations and VNR requirements. The problem of embedding VNR on the substrate network is called the *Virtual Network Embedding* (VNE) problem [1].

Typically, the main requirements of VN's nodes and links are respectively processing power and bandwidth. They have been usually assumed to be exactly known fix parameters beforehand [2]. However, on one hand, determining the exact amount of these requirements is quite difficult due to variable bit rate traffic of network applications. On the other hand, they may fluctuate over time [3] and even follow specific patterns such as cycle-stationary traffic pattern [4]. Generally, to guarantee QoS, these requirements are overestimated that increases InP cost. Therefore, in this paper, the focus is centered on investigating the VNE problem with the aim of minimizing embedding cost wherein bandwidth requirements of virtual links are *uncertain*. The uncertainty implies that the exact amounts of the required bandwidths are not known when a VNR is given to InP, and also may change over time periods. However, at the beginning of each period, when actual traffic is injected into the VN, the exact amounts of the bandwidths will be known. We assume that the uncertain bandwidth requirements follow known probability distributions.

Very little research has been conducted on the VNE problem considering the uncertainty of VNR requirements [2, 3, 5]. In [5], only the normal distribution of uncertain parameters was considered. Using the Robust Optimization Framework in [2, 3], conservative solutions are obtained that increase the OPEX of the InP in terms of substrate network resource consumption. In this paper, we use Stochastic Programming Framework to obtain a non-conservative solution to the problem.

1.1. Research Objective

This paper aims to answer the following research question: "A VNR with exact processing requirements of virtual nodes is given. The bandwidth requirements of each virtual link can change over time and is described by a specific probability distribution. What is the embedding of the VNR to obtain the lowest embedding cost over time?" This problem is called the *stochastic VNE* problem.

1.2. Contributions

Here, by assuming that a) migrating the mapped virtual nodes is not practical, and b) the required bandwidth of each virtual link is described by a known

probability distribution, we formulate the stochastic VNE problem in the form of two-stage mixed integer linear programming model. In this formulation, VNR embedding is conducted in two stages. In the first stage, the mapping of the virtual nodes is determined. This mapping is coordinated with link mapping, i.e., it is performed in such a way that not only the virtual node mapping cost but also, the *average cost* of mapping of virtual links is minimized. The second phase is performed when the exact amounts of required bandwidths are specified in each period, and only (re)maps virtual links. By changing bandwidth requirements over time, only the links mapping will be changed because the mapping of virtual nodes in the first stage has been made with respect to the probability distribution of random variables describing the required bandwidth of the virtual links.

We use Sample Average Approximation (SAA) technique [6] to solve the stochastic VNE problem. Each sample is consists of one or more scenarios where a scenario represents a realization of the random variables. Increasing the number of generated scenarios makes the solution to be more accurate; however, it leads to a large Mixed Integer Linear Programming (MILP) problem which is not tractable by the commercial solvers. To improve the scalability of the technique, we utilize the Benders method [7] and decompose the problem into two smaller and tractable sub-problems. Despite of improving the scalability, the Benders method may increase the problem solving time in comparison with classical techniques e.g., the Branch & Bound method [8]. For this reason, we propose a novel iterative algorithm that significantly reduces the problem solving time via four new acceleration techniques applied on the Benders method.

In summary, the major contributions of this paper are as follows:

- For the first time, the stochastic VNE problem is formulated using Stochastic Programming Framework.
- By exploiting the available information about the statistical distribution of the required bandwidths via the SAA technique, we achieve a non-conservative solution of the problem.
- To tackle the complexity raised by increasing the sample size, the problem is decomposed into two problems using the Benders decomposition algorithm.
- In order to reduce the problem solving time, a new algorithm is proposed to accelerate the Benders method; it reduces the solving time by 7.5 times compared with the Branch & Bound method.

1.3. Paper Organization

The rest of this paper is organized as follows. In Section 2, the VNE problem and related research are reviewed in terms of uncertainty modeling and solution methods. System model and problem formulation are discussed in Section 3, where InP decision-making procedure for each stage is expounded. Section 4 focuses on the solution method; decomposition of the problem based on the Benders algorithm is discussed. Then, in Section 5, we propose techniques to accelerate the decomposition, and develop our algorithm based on them. Section 6 is devoted to evaluate the performance of the proposed solution algorithm. Finally, Section 7 concludes this paper.

2. Related Research

Network virtualization has been considered as a key enabler in the cloud-computing paradigm. It plays a crucial role in enabling infrastructure resource sharing and providing IaaS services[9, 10].Nevertheless, when resource sharing is utilized many challenges including privacy and confidentiality of information[11–14], as well as the utilization of shared infrastructure resources [15] should be considered. In this section, we focus on research that has been devoted to the efficient allocation of infrastructure resources to customers' VNR requests. This section first focuses on studies wherein all requirements of VNR are certain; then, approaches to address uncertainty in the VNE problem are reviewed.

Many studies have been carried out concerning modeling and solution of the VNE problem [1]. Since it is a NP-Hard problem [16], [17], different techniques like column generation or heuristic algorithms lead to differences in solution time and quality. Regarding modeling, previous studies can be classified according to the a) online vs. offline position, b) requested resources, and c) InP objectives in VNR embedding. In online problems, as soon as a VNR is received, the decision is made regarding its embedding. However, in offline problems, InP makes a decision by considering a set of VNRs. In the following, previous research are reviewed regarding the aforementioned aspects.

In [18], VNR embedding is performed in two separate stages. First, virtual nodes are mapped to substrate nodes using a heuristic algorithm. After that, embedding of virtual links is carried out by reducing it to the Multi-Commodity Flow (MCF) problem. Since uncoordinated embedding of virtual nodes and links lead to a low quality solution, in [19], coordinated embedding of virtual links and nodes is considered. The authors assumed only a subset of substrate nodes can be used for mapping each virtual node, which are located within a certain radius from them. Considering this limitation, an augmented graph is constructed from substrate network; afterwards, the VNE problem is formulated using MILP framework as a MCF, and solved through LP relaxation. Finally, rounding techniques are applied to produce integer solutions. Authors in [20] constructed an augmented graph from substrate network by connecting each virtual node to the set of substrate nodes that satisfies the virtual node capacity constraint. Having this graph, the VNE problem is reduced to an MCF problem. Its LP relaxation is solved by column generation technique, and finally, using the B&B algorithm, integer solutions is derived. Similarly, in [21], also an augmented graph is constructed; and VNE is formulated as MCF. Using the dual variables of the restricted problem, the augmented graph links are weighted, and virtual links are mapped using the shortest paths between virtual nodes.

In [22], the authors considered the offline VNE problem and applied column generation technique to the relaxed version of the problem. Afterwards, integer solution was derived using the B&B algorithm and rounding technique. In [23], the authors first, constructed a graph from VNR wherein each node represents a candidate path for mapping a virtual link. A link between two nodes in the constructed graph means that it is possible to use both corresponding paths. Using the constructed graph, the authors transformed the VNE problem into the minimum cost maximum clique problem, and provided a heuristic solution. In [24], the authors focused on distributed VNE solution by suggesting a consensus-based auction mechanism that guarantees a $1 - \frac{1}{e}$ optimal approximation. Investigated studies have focused on wired networks while in [25], a

Table 1: Summary of Related Research

#Ref.	Solution Approach	Objectives	Resources	Online/Offline	Uncertainty	Uncertainty Approach
[18]	Heuristic	Revenue	BW & CPU	Online	—	—
[19]	LP relaxation-Rounding	Revenue-Load Balancing	BW & CPU	Online	—	—
[20]	LP relaxation-Column Generation-B&B	Cost	BW & CPU	Online	—	—
[21]	Feasible region reduction	Load Balancing	BW & CPU	Online	—	—
[22]	LP relaxation-Column Generation-B&B-Rounding	Revenue	BW & CPU	Offline	—	—
[23]	Finding the maximal clique	—	BW & CPU	Online	—	—
[5]	Heuristic	Cost	BW & Capacity	Online	BW (Normal RV.)	Stochastic link packing
[3]	Approximation	Utilization of the most congested link	BW	Online	BW (Bounded RV.)	Robust Optimization
[2]	Heuristic	Profit	BW & CPU	Offline	BW (Bounded RV.)	Chance Constrained Programming-Robust Optimization
This paper	Accelerated Benders Decomposition	Cost	BW & CPU	Online	BW (Bounded RV.)	Stochastic Programming-SAA

mobility aware virtual network embedding algorithm was presented.

Mentioned studies are summarized in Table 1. In the conducted research, the entire VNR requirements are assumed to be known. However, bandwidth and processing requirements may not be exactly known beforehand. Few studies have been carried out regarding uncertainty in VNE problem. In [5], uncertainty in virtual links bandwidth is modeled as normal random variables with known average and variance. Assuming the probability of physical link capacity limitation does not surpass a certain quantity, the problem was formulated as a stochastic link-packing problem. The authors constructed an augmented graph and modeled the problem as an MCF problem. They presented a heuristic algorithm wherein virtual nodes are initially mapped based on degree; and then, virtual links are mapped on the least costly paths. The authors in [3] modeled bandwidth uncertainty as a bounded random variable. The VNE problem is formulated as a robust optimization problem and its approximate version is solved.

In [2, 26], the required bandwidth of each virtual link was considered as a random variable whose values are taken from a symmetric interval. The authors initially presented a chance constraint formulation of the VNE problem wherein probability of violating bandwidth of substrate links and processing resources was limited to $1-\varepsilon$. The authors also presented Γ-robust formulation of the problem. Eventually, considering the hardness of the problem, a heuristic algorithm was proposed.

As summarized in Table 1, previous research mainly focused on uncertainty in the bandwidth of virtual links. In this paper, we also consider the bandwidth uncertainty which can be any bounded random variable that is different from [5] wherein uncertainty was modeled as a normal random variable. The main difference between the present study and articles [3] and [2], is using Stochastic Programming Framework to model the problem. Since Robust Optimization tries to optimize over worst-case condition [27], the solution obtained is more conservative than the optimal solution. In the present study, the stochastic VNE is formulated as a two-stage stochastic programming problem and by utilizing the SAA technique [6], a good approximate solution to the original problem is obtained.

3. System Model and Problem Formulation

In this section, we formulate the *stochastic VNE* problem. After explaining the assumptions, the decision making procedure is elaborated through an illus-

trative example, finally problem formulation based on the Stochastic Programming Framework is presented. The notations used in the paper are summarized in Table 2.

3.1. Assumptions

The stochastic VNE problem considered in this paper is embedding a VNR on the substrate network with the objective of minimizing total embedding costs. It is investigated in an online manner, i.e., each time the problem is solved, it decides on the mapping of a single VNR without any information about future requests. Inputs to the stochastic VNE problem are the substrate network and a VNR; the problem outputs are mapping of each virtual node to a physical node, and similar to [18], mapping of each virtual link on one or several paths in substrate network[1]. Similar to most studies carried out [1], each substrate node is used at most once for a VNR, and each node in a VNR can be embedded on any node in the substrate network.

Substrate network resources, including processing capacity and bandwidth, are assumed to be exactly known. Each virtual node has a known and fixed processing requirement to be fulfilled by one substrate node. We assume that migrating the embedded virtual nodes in the substrate network is not practical because significant configuration changes are needed in the substrate network; moreover, it disturbs the services provided inside the virtual network.

We say that virtual links required bandwidth is *time-varying* and *uncertain*, and make the following assumptions about it. The traffic in a VN, i.e., the required bandwidths of virtual links, can change over arbitrary number of time periods. The required bandwidths during each time period are fixed whose exact amounts are not known upon arrival of the VNR. However, they can be represented by *bounded continuous random variables* with known distributions, which are independent of each other. Later, when traffic is injected into the VN, the actual amount of the bandwidths are known[2] at the beginning of each period. An example of this traffic pattern is Cycle-Stationary Traffic [4].

3.2. Substrate Network Model

Substrate network is modeled as a directed graph $G = (V, E)$ wherein V represents the set of substrate network nodes and E stands as the set of substrate links. Each substrate node $i \in V$ has a specified processing capacity denoted by $Cap\,(i)$. Each directed link from node $i \in V$ to node $j \in V$ is modeled as a pair $(i, j) \in E$; its bandwidth capacity is denoted by $BW_{(i,j)}$.

3.3. VNR Model

Similar to the substrate network, a VNR is modeled as a directed graph $G^v = (V^v, E^v)$. Each virtual node $u \in V^v$ has a specified processing requirement shown as $CapD\,(u)$ which is a certain quantity. Bandwidth demand of each

[1] Though multi-path routing may leads to packet reordering, there are advantages here like a better use of substrate resources. Packet reordering effect can be substantially reduced utilizing hash based splitting and packet tagging [18].

[2] Uncertainty can be resolved through various events. For instance, the user presents a more accurate estimation of links bandwidth demands, or InP could wait for user traffic to enter the virtual network.

Table 2: Notations Summary

Parameters	
Substrate network nodes	V
Substrate network links	E
Virtual network nodes	V^v
Virtual network links	E^v
Cost of embedding $u \in V^v$ on $i \in V$ in a time period	$C_i^u \in \mathbb{R}_+$
Virtual nodes embedding cost matrix	$C = \left[C_i^u\right]_{\lvert V^v\rvert \times \lvert V\rvert}$
Cost of each unit of allocated bandwidth on link $(i,j) \in E$ in a time period	$D_{(i,j)} \in \mathbb{R}_+$
Virtual links embedding cost matrix	$D = \left[D_{(i,j)}\right]_{\lvert V\rvert \times \lvert V\rvert}$
Node $u \in V^v$ processing demand	$CapD\,(u) \in \mathbb{R}_+$
Processing capacity of node $i \in V$	$Cap\,(i) \in \mathbb{R}_+$
Available bandwidth of link $(i,j) \in E$	$BW_{(i,j)} \in \mathbb{R}_+$
Bandwidth matrix of substrate network	$BW = \left[BW_{(i,j)}\right]_{\lvert V\rvert \times \lvert V\rvert}$
Random variable represents bandwidth demand of link $(u,v) \in E^v$	$\widetilde{bw}_{(u,v)}$
Matrix of bandwidth random variables	$\widetilde{bw} = \left[\widetilde{bw}_{(u,v)}\right]_{\lvert V^v\rvert \times \lvert V^v\rvert}$
A realization of $\widetilde{bw}_{(u,v)}$	$bw_{(u,v)}$
A realization of matrix $\widetilde{bw}$	$bw = \left[bw_{(u,v)}\right]_{\lvert V^v\rvert \times \lvert V^v\rvert}$
Set of a generated samples of $\widetilde{bw}$	S
A realization of $\widetilde{bw}_{(u,v)}$ in sample $s \in S$	$bw_{(u,v)}^s$
Set of outgoing links from node u	$\delta^+(u)$
Set of incoming links to node u	$\delta^-(u)$
Variables	
$x_i^u = 1$ if $u \in V^v$ is mapped on $i \in V$	$x_i^u \in \{0,1\}$
Matrix of first stage decision variables	$X = \left[x_i^u\right]_{\lvert V^v\rvert \times \lvert V\rvert}$
Bandwidth allocated to link $(u,v) \in E^v$ on link $(i,j) \in E$	$y_{(i,j)}^{(u,v)} \in \mathbb{R}_+$
Matrix of the second stage decision variables	$Y_{\lvert V^v\rvert \times \lvert V^v\rvert \times \lvert V\rvert \times \lvert V\rvert}$
Bandwidth allocated on link $(i,j) \in E$ for link $(u,v) \in E^v$ in sample s	$y_{(i,j)}^{s,(u,v)} \in \mathbb{R}_+$
Dual variable corresponding to the first constraint in the SA-VNE-S	$\lambda_i^{s,(u,v)} \in \mathbb{R}$
Dual variable corresponding to the second constraint in the SA-VNE-S	$\mu_{(i,j)}^s \in \mathbb{R}_+$
The variable used in the Benders decomposition algorithm	$\rho \in \mathbb{R}$
Set of the known extreme points	P
Set of the known extreme directions	R

virtual link in a period follows a bounded continuous random variable with a known probability distribution such as uniform or bounded normal distribution. Matrix of random variables describing bandwidth of virtual links is shown by $\widetilde{bw}$ whose entry in row u and column v, i.e., $\widetilde{bw}_{(u,v)}$, is a random variable if link $(u, v) \in E^v$ exists, otherwise it is equal to zero. A realization of matrix $\widetilde{bw}$ in a period, i.e. a scenario, is denoted by bw.

3.4. Cost Model

Similar to [19], the objective of the VNE problem is to minimize the OPEX of InP which is comprised of the cost of mapping virtual nodes and links. C_i^u is the cost of mapping virtual node u on substrate node i which is a given parameter obtained based on the processing requirement u and resource i. $D_{(i,j)}$ is the cost of allocating a unit on bandwidth on the substrate link (i, j). The cost of mapping of a virtual link is the total cost of the allocated bandwidth for the link on paths in the substrate network, which is determined by the required bandwidth of the link and the routing of the paths. The exact formulation of the cost will be presented in the following in Section 3.7.

It should be note that we assume that the node and link mapping costs are time independent; i.e., C_i^u and $D_{(i,j)}$ are fixed values that does not change over time periods and the costs do not depend on the length of time periods.

3.5. Decision Making Procedure

Each VNR comprises of $G^v = (V^v, E^v)$, $CapD\,(u) \;\; \forall u \in E^v$, and $\widetilde{bw}$. InP first decides whether to accept or reject the request by investigating if the substrate network resources are sufficient to fulfill the requirements of the VNR. To do this, InP can replace each random variable with its maximum value and solve the problem deterministically, or use other methods such as [28], which is out of the scope of this paper.

In case the request is acceptable, then decisions for mapping the virtual nodes and links are made in two different stages: before and after resolution of the uncertainties. In the first stage, upon arrival of the VNR, when the exact values of the random variables are not yet specified, the decision is made for mapping of *virtual nodes*, which is optimized based on $CapD\,(u)$ and $\widetilde{bw}$. More precisely, InP solves the *stochastic VNE problem*, formulated in Section 3.7, to determine the mapping of virtual nodes in such a way that the total cost of virtual node mapping and the *average* cost of virtual link mapping is minimized. This node mapping which is coordinated by link mapping does not change over the time as long as the random variables, $\widetilde{bw}$, do not change. In the second stage, at the beginning of a time period when the uncertainty in the required bandwidth of virtual links is resolved, InP makes decisions of embedding *virtual links* on the substrate network paths. With a given node mapping, obtained in the first stage, virtual links embedding is obtained through solving the link embedding problem, formulated in Section 3.7, in polynomial time. In the case of time-varying traffic pattern, at the beginning of each time period, InP can change the link mapping by rerouting the path of virtual links in the substrate network according to the realized bandwidth in that period.

The reason behind this two-stage coordinated procedure is that node mapping is substantially different from link mapping both in theoretical and practical aspects. First, node mapping is a NP-Hard problem [19] while link mapping

can be performed in polynomial time via linear programming (LP). Second, as stated, migrating virtual nodes because of traffic changes is not straightforward due to practical challenges. However, modification of the paths of virtual links in the substrate network can be performed in a reasonable time by only changing the routing rules[3]. Third, most importantly, the node mapping is optimized with respect to the given probability distributions; therefore, it is a reasonable solution as long as the probability distributions do not change.

In summary, in the proposed approach, node mapping is not postponed until every realization of random variables since it is optimized based on the provided probability distributions and changing the mapping over the time is cost intensive; on the other hand, by postponing link mapping, we can exploit the accurate information in each realization for the optimal link mapping[4].

The decision-making procedure offered in this paper is different from previously mentioned articles in terms of the time interval between decision-making stages. In [2, 3, 5], links and nodes embedding are conducted in a conservative manner that leads to cost increase. In the proposed procedure, the mappings are conducted in two separated but coordinated phases. Whereas links embedding is not conducted in the first stage, it is not completely ignored. In the first stage, node mapping is coordinated with link mapping as InP minimizes the *average* cost of link mapping besides the cost of node embedding.

3.6. An Illustrative Example

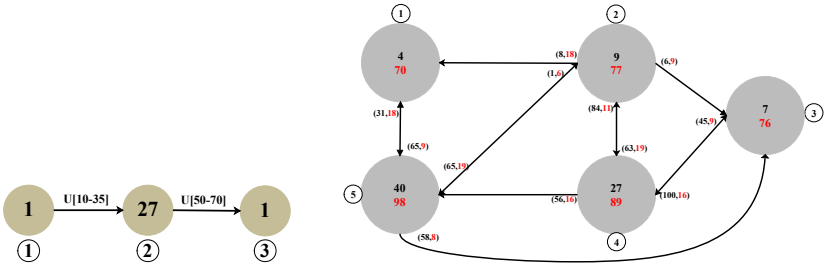
To clarify the assumptions and decision making procedure, in this section, an example is illustrated. Consider a situation wherein a user requests for a virtual network comprising of three nodes shown in Figure 1a. Traffic in the VN, changes over N time periods. Naturally, the user does not know the exact amount of required bandwidth for all the periods in advance. However, she/he can describe the required bandwidth for each link in each time period by a random variable. For example, if the random variable is $uniform\,[a-b]$, it means that the bandwidth of the link in the time period could be fixed amount between a and b based on the uniform distribution. In Figure 1a, assuming that the distributions does not change over time periods, the numbers shown on links indicate random variables describing bandwidth demand of the links, and the numbers inside filled circles express processing requirement of each virtual node. Numbers inside hollow circles show nodes indices.

Figure 1b shows the substrate network situation when InP receives the user's request. In this figure, there are two numbers inside each node. The first number in black indicates available processing resources, and the second number in red specifies the cost of a unit of utilized processing resource[5]. As an example, node

[3]Technologies like SDN facilitate path modification even more via programming interfaces to modify forwarding rules.

[4]Please note that the mapping of the nodes can be modified whenever the user traffic behavior is changed and cannot be described with the given probability distributions $\widetilde{bw}$ or in the case of changes in substrate network resources, e.g. node/link failure. In these cases, InP can apply the proposed algorithm in this paper to reoptimize node mapping based on the new condition however at the cost of virtual node migration. This is out of the scope of this paper.

[5]In this example, C_i^u is equal to required processing units of node u multiplied by the cost of a unit in node i.



(a) Virtual network request (b) Substrate network

Figure 1: An example of a VNR and a substrate network.

1 possesses four processing units each costs 70 units. Capacity and cost of each link in each direction are written above the link and near the source node of the link. For instance, the capacity of link (1, 5) is 31 units and the cost of allocating a unit of bandwidth on the link is 18, and link (5, 1) has a capacity of 65 unit with the bandwidth unit cost of 9.

As mentioned, for admission control, InP can solve the deterministic embedding problem using the maximum value of random variables, i.e., 35 and 70. In this example, the problem is feasible and its optimal objective value, i.e., the cost of mapping, as explained in Section 3.4, that encompasses the cost of allocating required processing units and the over estimated required bandwidth, is 4061. Therefore, InP knows that it can fulfill the VNR. As per the solution obtained, InP could conduct embedding of the VNR right at this stage. In this case, InP does not consider the uncertainty and the solution is called the *conservative solution*.

Considering the fact that the random variables describe required bandwidth for N time periods in the future, InP can conduct node mapping, but postpones link mapping until realization of the random variables when the exact amount of required bandwidth in each period is specified. In this decision making procedure, InP has three alternative approaches to map the virtual nodes:

1. *Worst-case* approach where InP conducts node mapping based on the conservative solution.
2. *Average-case* approach where InP carries out node mapping using a deterministic VNE problem in which every random variable is replaced by its average value, i.e., 22.5 and 60 in this example.
3. *Stochastic* approach where instead of using the average values of the random variables, the probability distribution functions are used to generate scenarios. InP conducts node mapping by solving the *stochastic VNE problem*, formulated in Section 3.7, that minimizes the cost of node mapping and the *average cost* of link mapping based on the generated scenarios.

Applying these approaches in this example leads to the node mappings shown in Table 3.

With these given node mappings, virtual links are mapped in each period by solving the *Link Embedding* problem formulated in Section 3.7. For 10 realizations of the required bandwidths, $N = 10$ time periods, the embedding cost, described in Section 3.4, are shown in Table 4. As evident, node mapping

Table 3: Node mapping by different approaches in Figure 1

Virtual Node	Mapped Substrate Node		
	Worst-case	Average-case	Stochastic-case
1	2	4	4
2	4	5	5
3	3	3	1

Table 4: The total cost of embedding in Figure 1 using different node mapping approaches

N	Requested bandwidth		Worst case	Average case	Stochastic
	(1,2)	(2,3)			
1	14	58	3638	**3499**	3551
2	24	67	3892	3965	**3848**
3	30	54	3750	**3723**	3771
4	11	62	3669	3563	**3539**
5	62	69	**4034**	4217	4082
6	31	65	3937	3985	**3886**
7	25	54	3695	**3643**	3691
8	14	57	3622	**3491**	3542
9	30	65	3926	3969	**3870**
10	30	70	**4006**	4199	4055
Average Cost	-	-	3816.9	3825.4	**3783.5**

based on the average-case solution leads to less cost than other approaches in periods 1, 3, 7 and 8. This situation is also the case for the worst-case solution in periods 5 and 10. On the contrary, if InP conducts node mapping according to the stochastic approach, shall have less cost in periods 2, 4, 6, and 9. More importantly, the average cost of 10 periods by the stochastic approach is less than the others. This shows that approaches that do not pay attention to the distribution of random variables, and only consider the upper limit have a more conservative behavior and therefore, their average cost will be higher than the stochastic approach. Finally, as indicated, the average cost of InP in all three approaches is less than the conservative solution i.e., 4061, which shows the advantage of the two-stage decision making procedure.

Note that although in the stochastic approach, the mapping of links and nodes is performed in two different stages, but since the mapping of the nodes is done according to the generated scenarios of possible realizations of bandwidth requirement, it is coordinated with link mapping. In fact, if there is little difference between the requested bandwidth in the future and the generated scenarios, the solution obtained by the stochastic approach would be very close to the optimal solution that has the lowest average cost over time. This can be achieved by increasing the number of generated scenarios. However, it increases computational complexity that will be discussed in the following sections.

3.7. Problem Formulation

Using Stochastic Programming Framework, the stochastic VNE problem is formulated as a two-stage stochastic mixed integer linear programming problem. In the first and second stages, respectively, decisions are made for node and link mapping. Formulation of the stochastic VNE problem is given below.

The objective function is the minimization of the total virtual nodes embedding cost and the average cost of virtual links embedding.

Stochastic VNE Problem:

$$\tilde{Z} = \min_{X} \sum_{i \in V} \sum_{u \in V^v} C_i^u x_i^u + E\left[Q\left(X, \widetilde{bw}\right)\right]$$

s.t.

$$\sum_{i \in V} x_i^u = 1 \quad \forall u \in V^v \tag{1}$$

$$\sum_{u \in V^v} x_i^u \leq 1 \quad \forall i \in V \tag{2}$$

$$CapD\left(u\right) x_i^u \leq Cap\left(i\right) \quad \forall u \in V^v,\ \forall i \in V \tag{3}$$

$$x_i^u \in \{0,1\} \quad \forall i \in V, \forall u \in V^v$$

The first constraint ensures that all virtual nodes are mapped on the substrate network. Constraint (2) enforces that no two nodes from a VNR are embedded on the same substrate network node. The third constraint guarantees that substrate node processing resources fulfill processing requirement of the virtual node mapped on it. $E\left[Q\left(X, \widetilde{bw}\right)\right]$ is the average cost of the second stage decisions, i.e., virtual links embedding, and stands for the expectation of function $Q\left(X, \widetilde{bw}\right)$ with respect to the probability distribution of $\widetilde{bw}$. The $Q\left(X, bw\right)$ is the optimal value of the second stage decisions with respect to the first stage decision variables X and a realization of uncertain values; it is defined as follows:

Link Embedding Problem:

$$Q\left(X, bw\right) = \min_{Y} \sum_{(i,j) \in E} \sum_{(u,v) \in E^v} D_{(i,j)} y_{(i,j)}^{(u,v)}$$

s.t.

$$\sum_{(i,j) \in E} y_{(i,j)}^{(u,v)} - \sum_{(j,i) \in E} y_{(j,i)}^{(u,v)} = bw_{(u,v)} \left(x_i^u - x_i^v\right) \tag{4}$$

$$\forall \left(u,v\right) \in E^v, \forall i \in V$$

$$\sum_{(u,v) \in E^v} y_{(i,j)}^{(u,v)} \leq BW_{(i,j)} \quad \forall \left(i,j\right) \in E \tag{5}$$

$$y_{(i,j)}^{(u,v)} \in \mathbb{R}_+ \quad \forall \left(u,v\right) \in E^v, \forall \left(i,j\right) \in E$$

Constraint (4) states that if x_i^u is equal to 1, bandwidth of virtual link (u, v), denoted by $bw_{(u,v)}$, is reserved on the outgoing links from node $i \in V$. Alternatively, If x_i^v is equal to 1, the bandwidth is reserved on incoming links to node $i \in V$. Finally, constraint (5) expresses that the total bandwidth of virtual links mapped on a physical link must not surpass its available bandwidth.

It should be noted that for the sake of simplicity, here, the stochastic VNE problem is formulated only for one time period. However, since the mapping of nodes, x_i^u, and distribution of the random variables do not change over time and the costs of using substrate resources, C_i^u and $D_{(i,j)}$, are time independent, $E[Q(X, \widetilde{bw})]$ can also be interpreted as expectation of $Q(X, bw)$ over multiple time periods.

In general, there are some challenges in calculating the quantity of $E\left[Q\left(X, \widetilde{bw}\right)\right]$, e.g., for continuous distributions, accurate value of the expectation cannot be

evaluated numerically [29]. Therefore, we resort to approximate the quantity of $E\left[Q\left(X,\widetilde{bw}\right)\right]$ rather than direct calculating it. One of the common methods is the SAA technique [6], which comprises of two stages as follows:

1. Generate a sample set S that includes $|S|$ scenarios. Each scenario is a realization of matrix $\widetilde{bw}$.
2. Instead of solving the stochastic VNE problem, solve the following problem:

$$\min_{X} \sum_{i\in V}\sum_{u\in V^v} C_i^u x_i^u + \frac{1}{|S|}\sum_{s\in S} Q\left(X, bw^s\right)$$
$$\text{s.t.}$$
$$(1),(2),(3)$$

Defining variable $y_{(j,i)}^{s,(u,v)}$ as the value of $y_{(j,i)}^{(u,v)}$ in scenario s, and $bw_{(u,v)}^s$ as the realization of the random variable $\widetilde{bw}_{(u,v)}$ in scenario s, $Q\left(X, bw^s\right)$ can be rewritten as follows, which is called the Second-Stage problem.

Second-Stage Problem:

$$Q\left(X, bw^s\right) = \min_{Y} \sum_{(i,j)\in E}\sum_{(u,v)\in E^v} D_{(i,j)} y_{(j,i)}^{s,(u,v)}$$
$$\text{s.t.}$$
$$\sum_{(i,j)\in E} y_{(i,j)}^{s,(u,v)} - \sum_{(j,i)\in E} y_{(j,i)}^{s,(u,v)} = bw_{(u,v)}^s\left(x_i^u - x_i^v\right) \quad (6)$$
$$\forall (u,v)\in E^v, \forall i \in V$$
$$\sum_{(u,v)\in E^v} y_{(i,j)}^{s,(u,v)} \le BW_{(i,j)} \quad \forall (i,j)\in E \quad (7)$$
$$y_{(i,j)}^{s,(u,v)} \in \mathbb{R}_+ \quad \forall (u,v)\in E^v, \forall (i,j)\in E$$

On this basis, the stochastic VNE problem can be rewritten as below titled as Sample Averaged VNE (SA-VNE) problem:

SA-VNE Problem:

$$\hat{Z} = \min_{X,Y} \sum_{i\in V^s}\sum_{u\in V^v} C_i^u x_i^u + \tfrac{1}{|S|}\sum_{s\in S}\sum_{(i,j)\in E^s}\sum_{(u,v)\in E^v} D_{(i,j)} y_{(j,i)}^{s,(u,v)}$$
$$\text{s.t.}$$
$$(1),(2),(3),(6),(7)$$
$$x_i^u \in \{0,1\} \quad \forall i\in V, \forall u \in V^v$$
$$y_{(i,j)}^{s,(u,v)} \in \mathbb{R}_+ \; \forall (u,v)\in E^v, \forall (i,j)\in E, \; \forall s \in S$$

Since the VNE problem in the deterministic case is an NP-Hard problem [19], therefore, the SA-VNE is NP-Hard as well. In the next section, we propose a solution to deal with its complexity.

4. Solution Methodology

It could be shown that by increasing the sample size, the optimal solution of the SA-VNE, $\hat{Z}$, and the first stage decision variables, X, converge with

probability one, to their true values obtained from solving the stochastic VNE problem [6]. However, increasing the number of scenarios leads to increasing the size of the SA-VNE. To improve the scalability in solving the problem, the Benders decomposition method is utilized, and SA-VNE is decomposed into two problems. To clarify it, a brief summary of the Benders method is presented; then, the decomposition of the SA-VNE problem is explained based on it.

4.1. Benders Decomposition Method

We describe the Benders decomposition method based on the following general problem called the *original problem*:

$$\begin{aligned} & \min_{X,Y} c^T X + b^T Y \\ \text{s.t.} \quad & \textbf{(original problem)} \\ & AX \geq d \\ & BX + DY \geq h \\ & X \in \mathbb{X},\ Y \geq 0 \end{aligned}$$

where $\mathbb{X} \subseteq \mathbb{R}^m$ and may contain integer variables and $Y \in \mathbb{R}^n$ and contains only continuous variables. In this formulation, all quantities are certain, and there is no uncertain value. In the Benders decomposition method, the original problem is broken into two problems, i.e., the master problem and the sub-problem. The sub-problem is defined as follows:

$$\begin{aligned} & \min_{Y} b^T Y \\ \text{s.t.} \quad & \textbf{(sub-problem)} \\ & \Pi : DY \geq h - B\bar{X} \\ & Y \geq 0 \end{aligned}$$

wherein $\bar{X}$ is obtained from the master problem solution and Π represents the dual variables corresponding to the sub-problem constraints. The dual of the sub-problem is:

$$\begin{aligned} & \max_{\Pi} \Pi^T \left(h - B\bar{X}\right) \\ \text{s.t.} \quad & \textbf{(dual of sub-problem)} \\ & D^T \Pi \leq b \\ & \Pi \geq 0 \end{aligned}$$

The master problem is:

$$\begin{aligned} & \min_{X,\rho} c^T X + \rho \\ \text{s.t.} \quad & \textbf{(master problem)} \\ & AX \geq d \\ & \bar{\Pi}_{opt}^T (h - BX) \leq \rho \quad \forall \bar{\Pi}_{opt}^T \in Extreme\_Points \\ & \textbf{(Optimality Cut)} \\ & \bar{\Pi}_{feas}^T (h - BX) \leq 0 \quad \forall \bar{\Pi}_{feas}^T \in Extreme\_Directions \\ & \textbf{(Feasibility Cut)} \\ & X \in \mathbb{X},\ \rho, \in \mathbb{R} \end{aligned}$$

where set $Extreme\_Points$ includes all extreme points and set $Extreme\_Directions$ includes all extreme-direction defined by the feasible region of the dual of the sub-problem. Since identifying all members of these sets is not initially possible, in the Benders method, the members are gradually identified in an iterative manner. Instead of solving the master problem, the restricted master problem is solved which is

$$\begin{aligned} &\min_{X,\rho} c^T X + \rho \\ \text{s.t.} \quad & \textbf{(restricted master problem)} \\ & AX \geq d \\ \bar{\Pi}_{opt}^T (h - BX) \leq \rho \quad & \forall \bar{\Pi}_{opt}^T \in P \quad \textbf{(Optimality Cut)} \\ \bar{\Pi}_{feas}^T (h - BX) \leq 0 \quad & \forall \bar{\Pi}_{feas}^T \in R \quad \textbf{(Feasibility Cut)} \\ & X \in \mathbb{X},\ \rho \in \mathbb{R} \end{aligned}$$

In which, P is a subset of the extreme-points and R is a subset of the extreme-directions which are constructed as follows.

After solving the restricted master problem, its solution, $\bar{X}$, is substituted in the dual of the sub-problem. If it turns out to be feasible and bounded, the dual vector Π^T is added to set P, which is denoted by $\bar{\Pi}_{opt}^T$. Otherwise, if the dual of the sub-problem turns out to be unbounded, then the sub-problem is infeasible. In this case, an extreme direction is derived and added to set R, which is shown by $\bar{\Pi}_{feas}^T$.

Assuming that the original problem is feasible and has a bounded optimal value, the Benders algorithm for solving it is shown in Algorithm 1. In this algorithm, $\mathcal{UB}$ and $\mathcal{LB}$ are respectively the upper and lower bounds for the optimal value of the original problem and k is a counter. The termination condition in Algorithm 1 is defined in terms of the difference between the upper bound and the lower bound. However, if the optimality is not the main concern, then other termination conditions, such as limitation on the number of times the while loop is executed, can be used to obtain sub-optimal approximated solution.

Since the number of extreme points and extreme directions in any polyhedron is finite, this algorithm generates a limited number cuts and eventually converges to the *optimal value* of the original problem. To prove correctness and convergence, readers may refer to [7].

4.2. Decomposition of the SA-VNE problem

In this section, we decompose the SA-VNE problem according to the Benders method. More precisely, the "SA-VNE problem," which correspond to the "original problem" in the Benders method, is decomposed into the "SA-VNE-RM," and "SA-VNE-S" problems which respectively correspond to the "restricted master problem," and "sub-problem" in the method. As explained, these problem are linked and solved iteratively via substituting the solution of the restricted master problem in the sub-problem and adding the cuts obtained from the "dual of sub-problem" to the "restricted master problem." In the SA-VNE case, the dual of the sub-problem is the "SA-VNE-DS." These problems are formulated in the followings.

Algorithm 1 Benders Decomposition Algorithm

$\mathcal{UB} \leftarrow \infty,\ \mathcal{LB} \leftarrow -\infty,\ k \leftarrow 0,\ P_0 \leftarrow \emptyset,$
$R_0 \leftarrow \emptyset, Termination\_Condition \leftarrow (\mathcal{UB} - \mathcal{LB} \leq \varepsilon)$

While:
$k \leftarrow k + 1$;
Solve the restricted master problem, and
    $\mathcal{LB} \leftarrow$ optimal objective function value
    $\bar{X} \leftarrow$ optimal decision variables

DSP:
Use $\bar{X}$ to solve the dual of the sub-problem.
if the dual of the sub-problem is unbounded **then**
    Save the returned $\bar{\Pi}^T_{feas}$
    $R_k \leftarrow R_{(k-1)} \cup \{\bar{\Pi}^T_{feas}\}$ and go to ***While***.
else
    Save optimal objective Z^*_{DSP} and decision variables $\bar{\Pi}^T_{opt}$
    $P_k \leftarrow P_{k-1} \cup \{\bar{\Pi}^T_{opt}\}$
    if $Z^*_{DSP} + c^T\bar{X} < \mathcal{UB}$ **then**
        $\mathcal{UB} \leftarrow Z^*_{DSP} + c^T\bar{X}$
        Calculate $\bar{Y}$ using $\bar{\Pi}^T_{opt}$
    if the $Termination\_Condition$ is satisfied **then**
        Go to ***Exit***
    else
        Go to ***While***.

Exit:
Return $\bar{X}$, $\bar{Y}$ as the optimal solution and exit.

The SA-VNE-S problem is as follows:

SA-VNE-S Problem:

$$\min_{Y} \frac{1}{|S|} \sum_{s \in S} \sum_{(i,j) \in E} \sum_{(u,v) \in E^v} D_{(i,j)} y^{s,(u,v)}_{(j,i)}$$

s.t.

$$\lambda^{s,(u,v)}_i : \sum_{(i,j) \in E} y^{s,(u,v)}_{(i,j)} - \sum_{(j,i) \in E} y^{s,(u,v)}_{(j,i)}$$
$$= bw^s_{(u,v)} \left(\bar{x}^u_i - \bar{x}^v_i\right) \ \forall (u,v) \in E^v, \ \forall i \in V, \ \forall s \in S$$
$$\mu^s_{(i,j)} : \sum_{(u,v) \in E^v} y^{s,(u,v)}_{(i,j)} \leq BW_{(i,j)} \quad \forall (i,j) \in E, \ \forall s \in S$$
$$y^{s,(u,v)}_{(i,j)} \in \mathbb{R}_+ \quad \forall (u,v) \in E^v, \forall (i,j) \in E, \forall s \in S$$

In the SA-VNE-S, the first stage variables $\bar{x}^u_i$ is obtained by solving the SA-VNE-RM and substituted in the SA-VNE-S. $\lambda^{s,(u,v)}_i$ and $\mu^s_{(i,j)}$ are dual variables corresponding to the SA-VNE-S constraints. Note that the problem can be treated as $|S|$ independent problems, i.e., an LP per scenario.

The dual of the SA-VNE-S problem, which is used to obtain the optimality and feasibility cuts, is formulated as below:

SA-VNE-DS Problem:

$$\max_{\lambda, \ \mu} \frac{1}{|S|} \sum_{s \in S} \Big(\sum_{(u,v) \in E^v} \sum_{i \in V} \lambda^{s,(u,v)}_i bw^s_{(u,v)} \left(\bar{x}^v_i - \bar{x}^u_i\right)$$
$$- \sum_{(i,j) \in E} \mu^s_{(i,j)} BW_{(i,j)} \Big)$$

s.t.

$$D_{(i,j)} + \lambda^{s,(u,v)}_i - \lambda^{s,(u,v)}_j + \mu^s_{(i,j)} \geq 0$$
$$\forall (i,j) \in E, \forall (u,v) \in E^v, \forall s \in S$$
$$\lambda^{s,(u,v)}_i \in \mathbb{R}, \ \mu^s_{(i,j)} \in \mathbb{R}_+$$

Finally, the SA-VNE-RM, corresponding to the "restricted master problem"

in the Benders method, is defined as follows:

SA-VNE-RM Problem:

$$\min_{X,\,\rho} \sum_{i\in V}\sum_{u\in V^v} C_i^u x_i^u + \rho$$

s.t.

$$\sum_{i\in V} x_i^u = 1 \quad \forall u \in V^v$$

$$\sum_{u\in V^v} x_i^u \leq 1 \quad \forall i \in V$$

$$CapD\,(u)\, x_i^u \leq Cap\,(i) \quad \forall u \in V^v, \forall i \in V$$

$$\frac{1}{|S|}\sum_{s\in S}\Big(\sum_{(u,v)\in E^v}\sum_{i\in V}\bar{\lambda}_i^{s,(u,v)} bw_{(u,v)}^s\,(x_i^v - x_i^u) - \sum_{(i,j)\in E}\bar{\mu}_{(i,j)}^s BW_{(i,j)}\Big) \leq \rho \;\; \forall\,(\bar{\lambda},\bar{\mu}) \in P$$

$$\frac{1}{|S|}\sum_{s\in S}\Big(\sum_{(u,v)\in E^v}\sum_{i\in V}\bar{\lambda}_i^{s,(u,v)} bw_{(u,v)}^s\,(x_i^v - x_i^u) - \sum_{(i,j)\in E}\bar{\mu}_{(i,j)}^s BW_{(i,j)}\Big) \leq 0 \;\; \forall\,(\bar{\lambda},\bar{\mu}) \in R$$

$$x_i^u \in \{0,1\} \quad \forall i \in V,\ \forall u \in V^v$$

$$\rho \in \mathbb{R}$$

As discussed, instead of solving the SA-VNE directly, we solve the SA-VNE-RM and the SA-VNE-DS by using Algorithm 1. Whereas the solutions obtained from both approaches are the same, there are significant differences in running time and scalability of the approaches, which are clarified in the next section.

4.3. Advantages and Pitfalls of the Decomposition

As stated earlier, the solution of the SA-VNE approaches to the true solution of the stochastic VNE problem by increasing the number of scenarios. Nevertheless, the solution is not scalable, i.e., it may not be possible to solve SA-VNE directly because it gets a huge MILP problem that cannot be tackled by the commonly used B&B methods. The main advantage of the proposed decomposition is that the huge MILP is divided into a small MILP, i.e., the SA-VNE-RM problem, and multiple independent LP problems, i.e., the SA-VNE-S problem. Therefore, firstly, whereas SA-VNE-RM is still an NP-hard problem, however, its size, specially the number of binary variables, does not increase by enlarging the number of scenarios; hence due to the small size of the problem, not only B&B method can solve it efficiently, but also, problem specific solutions such as the explicit enumeration [30] method or specialized heuristic algorithm can used. Secondly, the SA-VNE-DS is a linear programming problem which can be solved using interior point methods such as the Karmarkar algorithm [31] in polynomial time. Thirdly, since SA-VNE-S for each scenario is independent of other samples, it can be solved for each sample in parallel that can greatly reduce the effect of increasing sample size.

Despite of the advantages, the Benders algorithm in the early iterations generates solutions that are far from the optimal solution. This is due to the

Table 5: The effect of the proposed methods on the convergence speed.

Technique	Reduction in the number of iterations	Reduction in the SA-VNE-RM solution time	Reduction in the number of times the SA-VNE-RM is solved
Omission of infeasible solutions	✓		
Initial heuristic solutions to the SA-VNE-RM			✓
Adding upper-bound to the SA-VNE-RM		✓	
Generating Pareto-optimal cuts	✓		

fact that the decomposition removes the linking constraint between the variables of the SA-VNE-RM and SA-VNE-S problems, i.e., constraint (6). In the next section, we develop four techniques to tackle the problem and reduce the convergence time of the Benders decomposition algorithm.

5. Accelerated Benders Decomposition

In this section, we present a novel algorithm to solve the SA-VNE problem more efficiently. In the design of this algorithm, four techniques are used to accelerate the Benders decomposition for SA-VNE which are summarized in Table 5, where the effects of these techniques on the convergence speed algorithm are clarified.

5.1. Omission of Infeasible Solutions

Decomposition of the SA-VNE removes the linking constraint between variables of stage one and two, i.e., constraint (6). As a result, substitution of the optimal decision variables of the SA-VNE-RM, $\bar{x}_i^u$, in the SA-VNE-S may make it an infeasible problem. In this case, a new feasibility cut is added to the SA-VNE-RM and then the problem is resolved; this causes an extra iteration in the algorithm. In order to avoid such kinds of situation, we try to identify truly infeasible solutions and remove them from the solution space of the SA-VNE-RM problem, which is achieved in two ways. First, two sets are defined as follows:

$$Inf_{deg}^{+} = \left\{(u,i) \mid u \in V^v, i \in V, \delta^{+}(u) > 0, |\delta^{+}(i)| = 0\right\}$$

$$Inf_{deg}^{-} = \left\{(u,i) \mid u \in V^v, i \in V, \delta^{-}(u) > 0, |\delta^{-}(i)| = 0\right\}$$

where $\delta^{+}(j)$ is the set of outgoing links from node j and $\delta^{-}(j)$ is the set of incoming links to node j. Set Inf_{deg}^{+} represents node pairs in which virtual node u has outgoing links, but physical node i lacks it. Similarly, Inf_{deg}^{-} describes pairs u and i in which node u has incoming link, while node i lacks it. In these cases, virtual node u cannot be mapped on substrate node i. Therefore, in the SA-VNE-RM problem, if $x_i^u \; \forall (u,i) \in \{Inf_{deg}^{+} \cup Inf_{deg}^{-}\}$ is determined to be 1, then the SA-VNE-S becomes infeasible, which must be avoided.

Second, we define bandwidth matrix bw_{min} based on matrix $\widetilde{bw}$ in which each element is replaced with its least possible value. Then, we construct the following sets:

$$Inf_{bw}^{+} = \left\{(u,i) \mid u \in V^v, i \in V, \beta^{+}(u, bw_{min}) > \beta^{+}(i, BW)\right\}$$

$$Inf_{bw}^{-} = \left\{(u,i) \mid u \in V^v, i \in V, \beta^{-}(u, bw_{min}) > \beta^{-}(i, BW)\right\},$$

where $\beta^+(j, bw)$ and $\beta^-(j, bw)$ are respectively the total bandwidth of the outgoing and incoming links of node j based the bandwidth matrix bw. Inf^+_{bw} is a set of node pairs (u, i) in which sum of the minimum required bandwidth of outgoing links of virtual node u is greater than available bandwidth of outgoing links of substrate node i. Set Inf^-_{bw} represents the similar concepts for incoming links. In these cases, virtual node u cannot be mapped on substrate node i. Therefore, we should avoid $x^u_i = 1 \; \forall (u, i) \in \{Inf^+_{bw} \cup Inf^-_{bw}\}$ in the SA-VNE-RM problem that makes the SA-VNE-S problem infeasible. Putting together all above defined sets, the following constraint is added to the SA-VNE-RM problem:

$$x^u_i = 0, \; \forall (u, i) \in \left\{Inf^+_{bw} \cup Inf^-_{bw} \cup Inf^+_{deg} \cup Inf^-_{deg}\right\} \quad (8)$$

5.2. Initial Heuristic Solutions to the SA-VNE-RM

In the proposed decomposition, the SA-VNE-RM solution converges to the optimal value in an iterative manner by gradually constructing the sets P and R via the cuts derived from solving the SA-VNE-DS, where $\bar{x}^u_i$ are substituted by the values obtained from the SA-VNE-RM solution in the previous iteration. To reduce the convergence time, we propose that instead of directly solving the SA-VNE-RM in each iteration, which is time consuming and provides only a single solution, at the beginning a heuristic algorithm can find a few solutions to the problem, which are independently substituted in the SA-VNE-DS to generate multiple cuts. The cuts are placed in sets P or R before executing the Benders algorithm that reduce the solution space of the SA-VNE-RM and consequently the solution time of the algorithm. Moreover, since generating cuts corresponding to each solution of the SA-VNE-RM is independent of other solutions, the SA-VNE-DS can be executed for all solutions in parallel.

In the following, we develop an algorithm to find a set of possible solutions to the SA-VNE-RM problem which is shown in Algorithm 2. Whereas we do not seek for the optimal solution in this algorithm, minimizing the cost of embedding improves the quality of the corresponding optimality cuts. Therefore, at the beginning, if it is possible to map node u on node i, which is checked in Line 4, its cost is estimated in Line 5; where

$$cost^+(u, i) = \sum_{(i,j) \in L^+_i(\delta^+(u))} D_{(i,j)}$$

in which, $L^+_i(n)$ is the set of n least-cost outgoing links of i according to D. $cost^-(u, i)$ is defined in the similar way for incoming links of i and u.

According to the estimated costs, three kinds of node mapping (the solution of the SA-VNE-RM) are conducted. In Lines 6–11 and Lines 12–17, virtual nodes are respectively processed in ascending and descending order of degree; and each node is mapped on the least cost physical node. In Lines 18–25, a few random mappings are obtained. Finally, in Line 26, the cut corresponding to each possible solution is obtained by solving the SA-VNE-DS.

Algorithm 2 Heuristic for Initial Solutions

$cost_{|V^v|\times|V|} \leftarrow \infty, k \leftarrow 1$
for all $i \in V$ **do**
  for all $u \in V^v$ **do**
    if $\left(\beta^+(u, bw_{min}) \leq \beta^+(i, BW)\right)$ **and** $\left(\beta^-(u, bw_{min}) \leq \beta^-(i, BW)\right)$ **and** $\left(\delta^+(u) \leq \delta^+(i)\right)$ **and** $\left(\delta^-(u) \leq \delta^-(i)\right)$ **then**
      $cost(u,i) \leftarrow cost^+(u,i) + cost^-(u,i) + C_i^u$

$L \leftarrow$Sort V^v in ascending order of $(\delta^+(u) + \delta^-(u))$
$X_1 \leftarrow 0$;
while L is not empty **do**
  $u \leftarrow \text{Pop}(L.head)$
  $i \leftarrow \underset{j}{\text{argmin}} \{cost(u,j)\}$
  $X_1(u,i) \leftarrow 1$

$L \leftarrow$Sort V^v in descending order of $(\delta^+(u) + \delta^-(u))$
$X_2 \leftarrow 0$;
while L is not empty **do**
  $u \leftarrow \text{Pop}(L.head)$
  $i \leftarrow \underset{j}{\text{argmin}} \{cost(u,j)\}$
  $X_2(u,i) \leftarrow 1$

---

while $k < MAX\_Iterations$ **do**
  $X_{k+2} \leftarrow 0$
  $L \leftarrow$Permuted list of elements of V
  while L is not empty **do**
    $u \leftarrow \text{Pop}(L.head)$
    $i \leftarrow \text{argrandom}\{cost(u,j) < \infty\}$
    $X_{k+2}(u,i) \leftarrow 1$
  $k \leftarrow k + 1$

Solve the SA-VNE-DS in parallel and generate feasibility or optimality cuts.

5.3. Adding Upper Bound to the SA-VNE-RM

In the Benders algorithm, for the optimality cut generated in each iteration, we have the following equations:

$$\frac{1}{|S|}\sum_{s\in S}\Big(\sum_{(u,v)\in E^v}\sum_{i\in V}\bar{\lambda}_i^{s,(u,v)}bw^s_{(u,v)}\left(x_i^v-x_i^u\right) - \sum_{(i,j)\in E}\bar{\mu}^s_{(i,j)}BW_{(i,j)}\Big)\le\rho \quad (9)$$

and

$$\mathcal{UB}\ge\mathcal{LB}=\sum_{i\in V}\sum_{u\in V^v}C_i^u x_i^u+\rho \quad (10)$$

from (9) and (10), we conclude:

$$\mathcal{UB}+\frac{1}{|S|}\sum_{s\in S}\sum_{(i,j)\in E}\bar{\mu}^s_{(i,j)}BW_{(i,j)}\ \ge \quad (11)$$
$$\frac{1}{|S|}\sum_{s\in S}\Big(\sum_{(u,v)\in E^v}\sum_{i\in V}\left(\bar{\lambda}_i^{s,(u,v)}bw^s_{(u,v)}+C_i^v\right)\left(x_i^v\right) - \left(\bar{\lambda}_i^{s,(u,v)}bw^s_{(u,v)}+C_i^u\right)\left(x_i^u\right)\Big)$$

Adding cut (11) in each iteration, apart from the optimality cut, leads to a better estimation of the upper bound value in the B&B algorithm [8], and consequently improves solution time. In most commercial solvers, the constraint can accelerate solving the SA-VNE-RM [19], [32].

5.4. Generating Pareto-optimal Cuts

In the Benders decomposition, when the "sub-problem" is degenerate, there are several optimal solutions to the "dual of sub-problem;" and consequently, different optimality cuts can be generated. It was shown that using the *Pareto optimal* cut, defined in the following way, leads to a significant reduction in the convergence time of the Benders algorithm [32]. We say that the cut generated by $\Pi^T_{opt,1}$ dominates the cut generated by $\Pi^T_{opt,2}$ if $\Pi^T_{opt,1}\left(h-BX^*\right) > \Pi^T_{opt,2}\left(h-BX^*\right)$, where X^* is the optimal solution to the original problem. A cut is called *Pareto optimal* if it is not dominated by any other cut.

Since X^* is not known until the end of the solution process, Magnanti and Wong designed a problem that leads to generating a Pareto-optimal cut [33] without knowing the X^*. However, it suffers from two drawbacks. First, the problem is dependent on the solution of the "dual of sub-problem," and Papadakos [34] proved that when the dual problem is solved sub-optimally, the Magnanti-Wong problem might become infeasible. As a solution, he proposed the independent Magnanti-Wong problem and proved that the cut obtained through its solution is Pareto-optimal. Second, the Magnanti-Wong problem is formulated based on valid core points[6], while it was shown that finding the corner points is not an easy task [34]. To overcome the issue, Papadakos defined a Magnanti-Wong Point and proved that the convex combination of a Magnanti-Wong point with any other point in the feasible region of the master problem is also a Magnanti-Wong Point [34].

[6]A core point is a point in the relative interior of the convex hull of the master problem feasible region.

Based on these findings, Papadakos proposed an algorithm to improve the Benders algorithm. In each iteration of the algorithm, a new Magnanti-Wong point is obtained from convex combination of the current point and the solution of the restricted master problem; this new point is used to solve the independent Magnanti-Wong problem, and consequently to generate an optimality cut. Then, the algorithm continues to solve the restricted master problem and the dual of the sub-problem. By inspiration of the Papadakos algorithm, we design our proposed algorithm in the next section.

5.5. Proposed Algorithm to Solve the SA-VNE

Putting all the proposed acceleration techniques together, the proposed algorithm to solve the SA-VNE problem is described in Algorithm 3. In Line 2, by using Algorithm 2, a set of possible solutions for SA-VNE-RM is generated. Afterwards, the SA-VNE-DS is solved per possible solution X_i, and its corresponding objective value is saved in $Z^*_{DSP,i}$. In Lines 3 to 5, the $\mathcal{UB}$ quantity is calculated and a Magnanti-Wong point is derived.

In continuation, the algorithm enters into the main while loop. In Line 7, the SA-VNE-RM with additional constraints (8) and (11) is solved and then the value of the $\mathcal{LB}$ and current solution $\bar{X}$ are updated. In Line 11, the SA-VNE-DS is solved. If the problem is unbounded, a feasibility cut is added to the SA-VNE-RM and algorithm returns to the beginning of the while loop. Otherwise, the $\mathcal{UB}$ value is updated. If the termination condition is satisfied, the algorithm ends. Otherwise, in Line 24 the value of the Magnanti-Wong point is updated. After that, the SA-VNE-DS is solved by using the updated Magnanti-Wong point to generate a Pareto-optimal cut. After that, the generated cut is added to the SA-VNE-RM.

The condition for termination of Algorithm 3 execution is the convergence of the $\mathcal{UB}$ and $\mathcal{LB}$ values. Given the time constraint, the termination condition can be stated in the terms of k, i.e. the number of times the algorithm generates a new solution.

6. Numerical Results

In this section, after explanation of the evaluation environments and parameters, the result obtained by solving the SA-VNE is investigated and compared with the solution of the worst-case and conservative problems. Then, the solution time of Algorithm 3 is compared with the B&B method.

6.1. Simulation Settings

The parameters used to generate substrate network topologies and VNRs are shown in Table 6. In the topologies, directed links between every two nodes were created with a probability of 0.5. Two substrate networks were used, named Network 1 and Network 2, which have seven and eight nodes, respectively. In each of the networks, processing and bandwidth resources were initialized using uniform random variables with intervals indicated in Table 6 by $U[x-y]$.

In order to assess the effect of uncertainty interval on solution quality and time, two VNR requests with the same random topology with four nodes were generated. The VNRs are different from each other in the random variables

Algorithm 3 The Proposed Algorithm

$\mathcal{UB} \leftarrow \infty,\ \mathcal{LB} \leftarrow -\infty,\ k \leftarrow 0,\ P_0 \leftarrow \emptyset,\ R_0 \leftarrow \emptyset,\ Termination\_Condition \leftarrow (\mathcal{UB} - \mathcal{LB} \leq \varepsilon)$

Use Algorithm 2 and generate a few X_js, solve the SA-VNE-DS in parallel, save the optimal value as $Z^*_{DSP,j}$, update P and R.
$\mathcal{UB} \leftarrow \min_j \{(C^T X_j + Z^*_{DSP,j})\}$
$Index \leftarrow \arg\min_j \{(C^T X_j + Z^*_{DSP,j})\}$
$MWP \leftarrow X_{Index}$

While:
$k \leftarrow k + 1$
Solve the SA-VNE-RM with additional constraints (8) and (11).
if It returns a bounded optimal value Z^*_{MP} **then**
    $\mathcal{LB} \leftarrow Z^*_{MP}$
Save the optimal solution $\bar{X}$ and go to ***SA-VNE-DS***.

SA-VNE-DS:
Use $\bar{X}$ to solve the SA-VNE-DS.
if SA-VNE-DS is an unbounded problem **then**
    Save the returned $\bar{\Pi}^T_{feas}$
    $R_k \leftarrow R_{(k-1)} \cup \{\bar{\Pi}^T_{feas}\}$ and go to ***While***.
else
    Save the returned optimal value Z^*_{DSP} and $\bar{\Pi}^T_{opt}$
    if $Z^*_{DSP} + C^T \bar{X} < \mathcal{UB}$ **then**
        $\mathcal{UB} \leftarrow Z^*_{DSP} + C^T \bar{X}$
        Calculate $\bar{Y}$ using $\bar{\Pi}^T_{opt}$
    if $Termination\_Condition$ is satisfied **then**
        Go to ***Exit***
    else
        Go to ***Core***.

Core:
$MWP \leftarrow 0.5MWP + 0.5\bar{X}$
Solve SA-VNE-DS using MWP, and set $P_k \leftarrow P_{(k-1)} \cup \{\bar{\Pi}^T_{opt}\}$
Go to ***While***.

Exit:
Return $\bar{X}, \bar{Y}$ as the optimal solution and exit.

Table 6: Simulation Parameters

Substrate Networks				
			Network 1	Network 2
Number of nodes			7	8
Link bandwidth (unit)			$U\,[1-100]$	
Processing resources (unit)			$U\,[1-50]$	
Processing cost (per unit)			$U\,[50-100]$	
Bandwidth cost (per unit)			$U\,[5-20]$	
Virtual Network Requests				
			VNR 1	VNR 2
Number of nodes			4	
Processing demand (unit)			$U\,[1-30]$	
Bandwidth demand (unit)			$U\,[1-70]$	$U\,[35-70]$
environments				
	Network 1	Network 2	VNR1	VNR2
environment 1	✓		✓	
environment 2	✓			✓
environment 3		✓	✓	

intervals used for required resources. As per Table 6, links bandwidth requirements in VNR 1 and VNR 2 respectively follow $U[1-70]$ and $U[35-70]$ random variables.

Three simulation environments, shown in Table 6, were designed to evaluate the performance of the algorithms. In the first environment, VNR 1 is mapped on Network 1. In the second environment, VNR 2 is mapped on Network 2, and in the third environment, VNR 1 is mapped on substrate Network 2. CVX version 2.1 with Mosek solver was used with their default parameters to solve the SA-VNE and the other problems. All the problems were solved to optimality and ε in Algorithm 3 was initialized to 10^{-3}. In Algorithm 2, 20 initial solutions were generated and optimality and feasibility cuts corresponding to these solutions were generated in parallel. The evaluation was carried out on a system with two cores operating at 3.3 GHz with 6 Gigabytes of memory.

6.2. Assessment of Solution Quality

In this section, the optimal value of the SA-VNE problem, which is obtained for a VNR in a time period[7], is compared with the solutions of the worst-case and the conservative problems. The conservative problem is a one-stage deterministic problem where each random variable is replaced with its maximum value. The objective function of the conservative problem is the total costs of virtual nodes and links mapping. In the worst-case problem, InP maps virtual nodes according to the solution of the conservative problem, but mapping of virtual links is postponed until the uncertainty is resolved. Therefore, the objective function of the worst-case problem is the total cost of virtual node mapping and the average cost of link mapping based on the generated scenarios. As a reminder, the objective function of the SA-VNE is the same as the worst-case problem.

The mean values and the standard deviations of the objective function of the problems in each environment are shown in Figure 2. Since the optimal

[7]In these simulations, the required bandwidths do not change over time, however, it is not known in the node embedding stage and realized after when the link embedding is performed.

objective value of the conservative problem is independent of the sample size and substantially differs from the optimal solution of the other algorithms, its optimal solution is separately depicted under the figures. As evident, there is a significant difference between the optimal values of these problems, and the optimal value of the SA-VNE is always less than the others. In addition, in all environments, as the sample size increases, the standard deviation values and fluctuations in the objective function values of the SA-VNE and the worst-case problems decrease.

In the first environment in Figure 2a, the optimal value obtained from the SA-VNE problem is on average up to 30% less than the conservative problem; it shows the advantage of the two-stage decision making procedure that exploits the available information of the realized bandwidth demands. Moreover, the optimal value of the SA-VNE is up to 2% less than the worst-case problem. In the SA-VNE, as the sample size increases from 10 to 500, the standard deviation is lowered from 224 to 27 that indicates that the solutions are getting closer to each other and to the true solution of the stochastic VNE problem.

When virtual links uncertainty interval is decreased, the difference between optimal values of the SA-VNE and the worst-case problems decreases too. This is the situation that occurs in the second environment, shown in Figure 2b, where the uncertainty interval is decreased from 70 to 35 in comparing to the first environment. It is evident that the optimal value of the SA-VNE shows a difference of 0.5% and 17% with the worst-case and conservative problems, respectively. In addition, by increasing the sample size from 10 to 500, the standard deviation of the SA-VNE is reduced from 179 to 21.

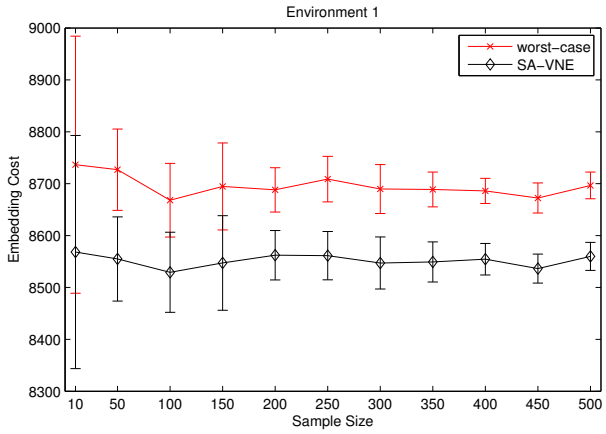
By increasing the size of the substrate network in the third environment, the difference between the optimal value of the SA-VNE and the other problems was increased as well. As illustrated in Figure 2c, the optimal value of the SA-VNE is lower up to 11% and 45% than the worst-case and the conservative problems, respectively. It shows the effectiveness of considering the average link mapping cost, in addition to the node mapping cost, in the first stage by the SA-VNE.

These noticeable difference between the SA-VNE in comparison with the worst-case problem stimulates InP to use its solution in spite of higher computational costs, which is also manageable as the result presented in the next section.

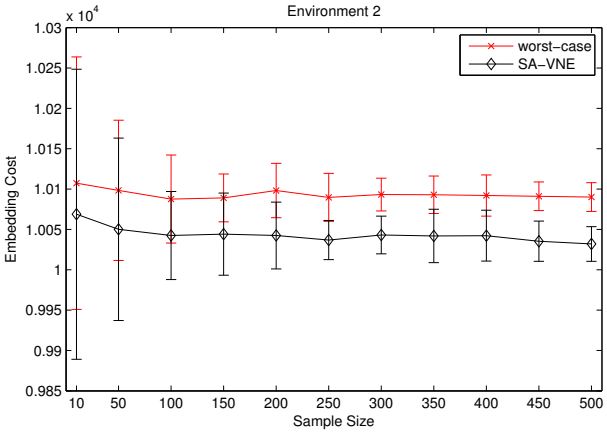
6.3. Assessment of Solution Time

Increasing the sample size improves the quality of the solution at the cost of increased solution time. Therefore, in this section, the solution time of the SA-VNE through Algorithm 3 is compared to the time needed to solve the problem by an off-the-shelf B&B based solvers. The running time of Algorithm 3 comprises the running time of the SA-VNE-RM, SA-VNE-DS, and Core sections plus the time spent on generating initial cuts by Algorithm 2. The solution time of the SA-VNE by Algorithm 3 and the B&B algorithm in all three environments is compared in Figure 3.

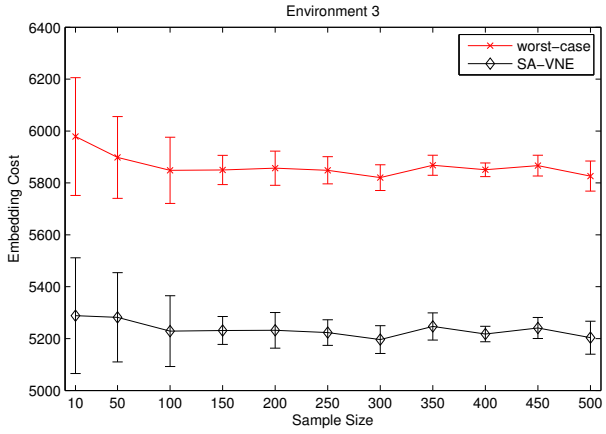
As evident, by increasing the sample size from 10 to 500, the running time of the B&B algorithm intensively increases. This increase is due to a sharp increase in the number of the second stage variables, which are *simultaneously* considered with the first stage variables in this algorithm. Although the computation time of the B&B algorithm for sample size less than 100 is lower than Algorithm 3,



(a) Conservative=12119

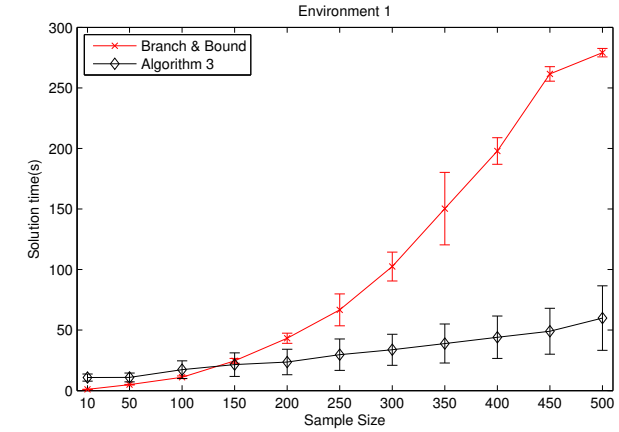


(b) Conservative=12119

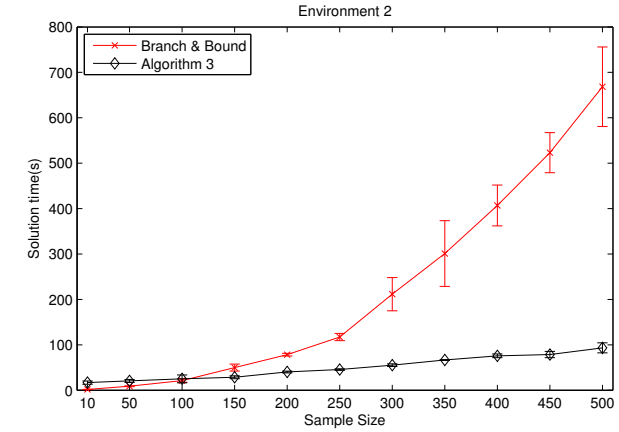


(c) Conservative=9592

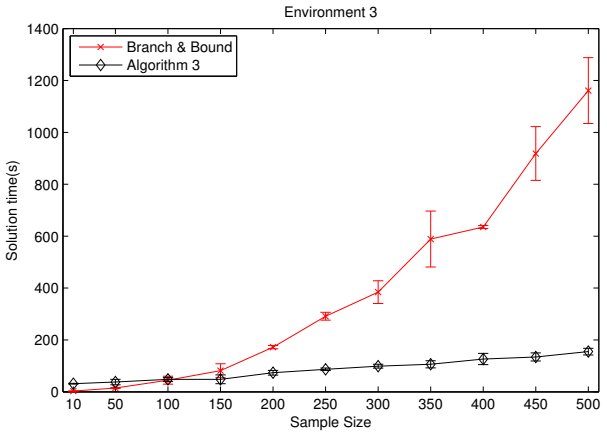
Figure 2: Embedding cost, i.e., objective function values, of different algorithms with respect to the sample size in each environment. Standard Deviations are shown in the form of error bars.



(a)



(b)



(c)

Figure 3: Solution time of the SA-VNE with respect to the sample size in different environments. Standard Deviations are shown in the form of error bars.

Table 7: Details of running time of Algorithm 3 in a sample run of environment 3.

$\|S\|$	Running Time (s)					
	Total	**SA-VNE-RM**	**SA-VNE-DS**	**Core**	**SA-VNE-RM / Iterations**	**SA-VNE-DS / Iterations**
10	32.1	10.8	10.9	10	0.2	0.2
50	49.9	10.7	19.6	18.9	0.2	0.4
100	53.4	8.8	22.7	21.1	0.2	0.6
200	74.6	7.7	35.5	30.3	0.2	1
300	96.3	6.9	45.5	42.4	0.2	1.5
400	106.8	6	52.9	45.8	0.2	2
500	169.9	8.1	86.2	72.5	0.3	2.7

but by increasing the sample size, it is observed that the computation time of Algorithm 3 increases with a mild slope and the difference between the solution times of the algorithms is increased. For instance, in the first environment, by multiplying the sample size by 50, the solution time of Algorithm 3 and the B&B become 4 and 276 times larger, respectively. Algorithm 3 in the first, second and the third environments is up to 4.7, 7.1 and 7.5 times faster than B&B, respectively. Putting the results in Figure 2 and Figure 3 together, indicates the proposed algorithm can achieve the solution, using a sufficient large sample size, in a reasonable time.

The breakdown of the running time of different sections of Algorithm 3 in a sample run of the environment 3 is shown in Table 7. In order to eliminate the effect of initial random solutions, the same set of initial solutions is used in all the sample sizes to generate the cuts. The total running time of Algorithm 3 is shown in column 2; it is seen that enlarging the sample size increase the running time. The total time is broken according to the sections of the algorithm in columns 3–5. As depicted, most of the time is spent in solving the SA-VNE-DS, which could be drastically reduced by solving it in parallel. More importantly, by increasing the sample size, the time spent on the master problem is remained almost constant since the decision variables of the problem, X, are independent of the sample size. Columns 6 and 7 show the average solution time of the SA-VNE-RM and the SA-VNE-DS in each iteration. The average solution time of the SA-VNE-RM does not change notably by increasing the sample size since it does not change the number of binary decision variables of the SA-VNE-RM, which are the main source of the complexity of the problem. The average solution time of the SA-VNE-DS, as expected, increases by enlarging the sample size.

7. Conclusion and Future Work

Uncertainty in the VNE problem is a research challenge that has received little attention. The present study considered uncertainty in virtual links bandwidth. To minimize embedding cost, it is aimed to find a non conservative solution by exploiting the statistical distribution of the required bandwidths via

formulating the problem as a two-stage MILP in the stochastic programming framework.

It was shown by simulation that the embedding obtained from the Stochastic VNE problem could reduce InP costs up to 40% compared with the conservative approaches. In order to solve the stochastic VNE problem, we utilized the sample average approximation technique, where by increasing the sample size, a more precise estimation of the solution to the stochastic VNE problem could be obtained. Nevertheless, it causes an increase in the problem size that intensively increases the computation time. To tackle the complexity, a novel algorithm based on the Benders decomposition method was proposed which could decrease the running time by 7.5 times compared to the branch and bound method.

There are still challenges remaining that can be studied in the future. We would like to improve the proposed heuristic algorithm to generate better initial solutions. An idea for this improvement can be the use of approximation algorithms. Generating feasible solutions in the early stages of the proposed algorithm leads to a faster convergence rate. Another point of interest to consider is the possibility of migrating embedded virtual nodes due to traffic changes in the stochastic VNE problem. This can lead to a lower embedding cost. However, the cost of migration and disruption in the service should also be considered.

References

[1] A. Fischer, J. F. Botero, M. T. Beck, H. de Meer, X. Hesselbach, Virtual Network Embedding: A Survey, IEEE Communications Surveys Tutorials 15 (4) (2013) 1888–1906, ISSN 1553-877X.

[2] S. Coniglio, A. Koster, M. Tieves, Data Uncertainty in Virtual Network Embedding: Robust Optimization and Protection Levels, Journal of Network and Systems Management 24 (3) (2016) 681–710, ISSN 1064-7570, 1573-7705.

[3] S. S. W. Lee, K. Y. Li, K. Y. Chan, Y. C. Chung, G. H. Lai, Design of bandwidth guaranteed OpenFlow virtual networks using robust optimization, in: Proc. IEEE Global Communications Conference, 1916–1922, 2014.

[4] V. Eramo, E. Miucci, M. Ammar, Study of Reconfiguration Cost and Energy Aware VNE Policies in Cycle-Stationary Traffic Scenarios, IEEE Journal on Selected Areas in Communications 34 (5) (2016) 1281–1297, ISSN 0733-8716.

[5] G. Sun, H. Yu, L. Li, V. Anand, Y. Cai, H. Di, Exploring online virtual networks mapping with stochastic bandwidth demand in multi-datacenter, Photonic Network Communications 23 (2) (2012) 109–122, ISSN 1387-974X, 1572-8188.

[6] A. J. Kleywegt, A. Shapiro, T. Homem-de Mello, The sample average approximation method for stochastic discrete optimization, SIAM Journal on Optimization 12 (2) (2002) 479–502.

[7] J. F. Benders, Partitioning procedures for solving mixed-variables programming problems, Numerische mathematik 4 (1) (1962) 238–252.

[8] D.-S. Chen, R. G. Batson, Y. Dang, Applied Integer Programming: Modeling and Solution, John Wiley & Sons, Inc., Hoboken, NJ, USA, ISBN 978-1-118-16600-0 978-0-470-37306-4, 2009.

[9] V. D. Piccolo, A. Amamou, K. Haddadou, G. Pujolle, A Survey of Network Isolation Solutions for Multi-Tenant Data Centers, IEEE Communications Surveys Tutorials 18 (4) (2016) 2787–2821, ISSN 1553-877X.

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

[11] D. Zissis, D. Lekkas, Addressing cloud computing security issues, Future Generation Computer Systems 28 (3) (2012) 583–592, ISSN 0167-739X.

[12] P. Li, T. Li, H. Ye, J. Li, X. Chen, Y. Xiang, Privacy-preserving machine learning with multiple data providers, Future Generation Computer Systems 87 (2018) 341–350, ISSN 0167-739X.

[13] J. Li, X. Chen, S. S. M. Chow, Q. Huang, D. S. Wong, Z. Liu, Multi-authority fine-grained access control with accountability and its application in cloud, Journal of Network and Computer Applications 112 (2018) 89–96, ISSN 1084-8045.

[14] J. Shen, T. Zhou, X. Chen, J. Li, W. Susilo, Anonymous and Traceable Group Data Sharing in Cloud Computing, IEEE Transactions on Information Forensics and Security 13 (4) (2018) 912–925, ISSN 1556-6013.

[15] S. S. Manvi, G. Krishna Shyam, Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey, Journal of Network and Computer Applications 41 (2014) 424–440, ISSN 1084-8045.

[16] D. G. Andersen, Theoretical approaches to node assignment, Tech. Rep., Carnegie Mellon University, 2002.

[17] E. Amaldi, S. Coniglio, A. M. C. A. Koster, M. Tieves, On the computational complexity of the virtual network embedding problem, Electronic Notes in Discrete Mathematics 52 (2016) 213–220, ISSN 1571-0653.

[18] M. Yu, Y. Yi, J. Rexford, M. Chiang, Rethinking Virtual Network Embedding: Substrate Support for Path Splitting and Migration, SIGCOMM Comput. Commun. Rev. 38 (2) (2008) 17–29, ISSN 0146-4833.

[19] M. Chowdhury, M. R. Rahman, R. Boutaba, ViNEYard: Virtual Network Embedding Algorithms with Coordinated Node and Link Mapping, IEEE/ACM Transactions on Networking 20 (1) (2012) 206–219, ISSN 1063-6692.

[20] Q. Hu, Y. Wang, X. Cao, Resolve the virtual network embedding problem: A column generation approach, in: 2013 Proceedings IEEE INFOCOM, 410–414, 2013.

[21] R. Mijumbi, J. Serrat, J. L. Gorricho, R. Boutaba, A Path Generation Approach to Embedding of Virtual Networks, IEEE Transactions on Network and Service Management 12 (3) (2015) 334–348, ISSN 1932-4537.

[22] A. Jarray, A. Karmouch, Decomposition Approaches for Virtual Network Embedding With One-Shot Node and Link Mapping, IEEE/ACM Transactions on Networking 23 (3) (2015) 1012–1025, ISSN 1063-6692.

[23] L. Gong, H. Jiang, Y. Wang, Z. Zhu, Novel Location-Constrained Virtual Network Embedding LC-VNE Algorithms Towards Integrated Node and Link Mapping, IEEE/ACM Transactions on Networking 24 (6) (2016) 3648–3661, ISSN 1063-6692.

[24] F. Esposito, D. D. Paola, I. Matta, On Distributed Virtual Network Embedding With Guarantees, IEEE/ACM Transactions on Networking 24 (1) (2016) 569–582, ISSN 1063-6692.

[25] G. Chochlidakis, V. Friderikos, Mobility Aware Virtual Network Embedding, IEEE Transactions on Mobile Computing 16 (5) (2017) 1343–1356, ISSN 1536-1233.

[26] S. Coniglio, A. M. C. A. Koster, M. Tieves, Virtual network embedding under uncertainty: Exact and heuristic approaches, in: 2015 11th International Conference on the Design of Reliable Communication Networks (DRCN), 1–8, 2015.

[27] D. Bertsimas, D. Brown, C. Caramanis, Theory and Applications of Robust Optimization, SIAM Review 53 (3) (2011) 464–501, ISSN 0036-1445.

[28] A. Blenk, P. Kalmbach, P. v. d. Smagt, W. Kellerer, Boost online virtual network embedding: Using neural networks for admission control, in: 2016 12th International Conference on Network and Service Management (CNSM), 10–18, 2016.

[29] A. Shapiro, Stochastic programming approach to optimization under uncertainty, Mathematical Programming 112 (1) (2008) 183–220, ISSN 0025-5610, 1436-4646.

[30] D. A. Tarvin, R. K. Wood, A. M. Newman, Benders decomposition: Solving binary master problems by enumeration, Operations Research Letters 44 (1) (2016) 80–85, ISSN 0167-6377.

[31] N. Karmarkar, A new polynomial-time algorithm for linear programming, Combinatorica 4 (4) (1984) 373–395, ISSN 0209-9683, 1439-6912.

[32] T. Santoso, S. Ahmed, M. Goetschalckx, A. Shapiro, A stochastic programming approach for supply chain network design under uncertainty, European Journal of Operational Research 167 (1) (2005) 96–115, ISSN 0377-2217.

[33] T. L. Magnanti, R. T. Wong, Accelerating Benders Decomposition: Algorithmic Enhancement and Model Selection Criteria, Operations Research 29 (3) (1981) 464–484, ISSN 0030-364X.

[34] N. Papadakos, Practical Enhancements to the Magnanti-Wong Method, Operations Research Letters 36 (4) (2008) 444–449, ISSN 0167-6377.