PAPER Special Section on Network Resource Control and Management Technologies for Sustainable Social Information Infrastructure

Reducing Dense Virtual Networks for Fast Embedding[*]

Toru MANO[†a)], *Nonmember*, Takeru INOUE[†], Kimihiro MIZUTANI[††], *and* Osamu AKASHI[†††], *Members*

SUMMARY Virtual network embedding has been intensively studied for a decade. The time complexity of most conventional methods has been reduced to the cube of the number of links. Since customers are likely to request a dense virtual network that connects every node pair directly ($|E| = O(|V|^2)$) based on a traffic matrix, the time complexity is actually $O(|E|^3 = |V|^6)$. If we were allowed to reduce this dense network to a sparse one before embedding, the time complexity could be decreased to $O(|V|^3)$; the time saving would be of the order of a million times for $|V| = 100$. The network reduction, however, combines several virtual links into a broader link, which makes the embedding cost (solution quality) much worse. This paper analytically and empirically investigates the trade-off between the embedding time and cost for the virtual network reduction. We define two simple reduction operations and analyze them with several interesting theorems. The analysis indicates that an exponential drop in embedding time can be achieved with a linear increase in embedding cost. A rigorous numerical evaluation justifies the desirability of the trade-off.

key words: *network virtualization, virtual network embedding, heuristics*

1. Introduction

Network virtualization [1]–[3] allows logically separated Virtual Networks (VNs) to coexist on a common Physical Network (PN), and provides network operators with flexibility, diversity, security, and manageability. In network virtualization, the role of traditional ISPs is divided into those of Service Providers (SPs) and Infrastructure Providers (InPs); an SP makes a request to an InP to build a VN, whereupon the InP embeds the VN in its PN. Virtual Network Embedding (VNE) is an optimization problem, in which the InP attempts to find a minimum cost embedding between the desired VN and its PN with respect to the constraints of physical resources (nodes and links) [3]. As this problem is known to be $\mathcal{NP}$-hard [4], many heuristics have been developed to identify better embedding arrangements [5]–[9].

Linear Programming Relaxation (LP Relax) is one of the most conventional and successful VNE approaches [9]–[12] as in other $\mathcal{NP}$-hard problems [13]. This approach formulates VNE as a mixed integer linear programming, which is then relaxed to obtain a fractional solution, and is finally rounded to a discrete solution. The time complexity, roughly proportional to the *cube* of the number of nodes and links, is a key factor in VNE along with the embedding cost. Of particular importance, it is often desired to be completed in a few minutes, since modern virtualized computing infrastructures, such as Amazon EC2, Google Compute Engine, and Microsoft Azure, have provisioning process cycles of less than ten minutes.

However, SP's demands for dense connections will increase the time complexity in practice, as will be explained below. Since SPs will focus on their own business, which is not network operation, they have no skill or motivation to "optimize" their VNs before submitting them to the InPs. That is, their VNs will simply reflect the estimated traffic matrix between virtual nodes [14], and the resultant VNs connect every node pair directly; i.e., the network topology is *complete* ($|E| = O(|V|^2)$). As a result, the VNE time could grow to $O(|V|^6)$; e.g., for $|V| = 100$, the time gap is a factor of one million, $|V|^6/|V|^3 = 10^6$.

InPs are, of course, allowed to redesign VNs to quickly embed them, as long as the original requirements are satisfied, but there has been, up to now, no paper that investigated the impact of VN redesign. Figure 1 shows an example; a triangle is reduced to a line. Since the required capacity of the removed link is pushed to the remaining two links, the new VN can handle any traffic carried on the original one. Although this operation successfully removes one link from the VN, the required link capacity is *doubled* on the remaining links. Through repeated VN reductions, the required link capacity may increase exponentially in the worst-case.

This paper studies a preprocessing scheme that converts a dense VN into a sparse one, so as to reduce the embedding time to $O(|V|^3)$ even for a VN of $|E| = O(|V|^2)$, without significantly degrading embedding cost. We assume that the preprocessing scheme is to be used by InPs, but it may also be used by SPs if VN density impacts the fee. Our contributions are summarized as follows:

- A new VNE operation class, *restriction*, is introduced, which converts a dense VN into a sparse one while preserving its embeddability (Sect. 3.1).
- Two metrics are also devised to measure the quality of restriction operation; *capacity ratio* is associated with embedding cost (objective value), while *feasibility ratio*

Manuscript received May 13, 2019.
Manuscript revised August 6, 2019.
Manuscript publicized October 25, 2019.

[†]The authors are with NTT Network Innovation Laboratories, Yokosuka-shi, 239-0847 Japan.
[††]The author is with Kindai University, Higashiosaka-shi, 577-8502 Japan.
[†††]The author is with National Institute of Informatics, Tokyo, 101-8430 Japan.

[*]A preliminary version of this paper was published in the *Proceedings of the 35th Annual IEEE International Conference on Computer Communications* (IEEE INFOCOM), San Francisco, CA, April 2016 (pp.1–9).

a) E-mail: toru.mano.sp@hco.ntt.co.jp

DOI: 10.1587/transcom.2019NRP0004

is related to feasibility (constraints) (Sect. 3.2).

- Two instances of restriction are presented; *node insertion* emulates hierarchical design by introducing core virtual nodes, while *link reduction* simply combines a virtual link with a neighboring virtual path (Sects. 3.3 and 4).
- Several interesting properties yielded by the restriction operations, such as tight bounds of the metrics, are proved. They imply a good trade-off: the exponential decrease of embedding time with only a linear increase in embedding cost (Sect. 5).
- Numerical evaluation confirms the trade-off with over ten thousand VNE instances (Sect. 6).

The public network is an essential part of the social information infrastructure since it connects a vast number of devices and enables high-speed and low latency communication among them. To enable the "Sustainable Social Information Infrastructure," it is vital to manage effectively as traffic is rapidly increasing; global IP traffic and the number of connected devices are expected to increase by three-fold and 50% in five years, respectively [15]. To meet these increases, network service providers have started to leverage network virtualization: existing studies propose a management framework [16] and report the implementation of a virtualization platform in a commercial environment [17]. This paper supports these movements by proposing the preprocessing scheme that aids the effective deployment of virtualized infrastructures on the physical network.

The rest of this paper is organized as follows. Section 2 formally defines VNE. Sections 3 and 4 defines restriction operations and introduces instances. Their theoretical properties are proved in Sect. 5. Section 6 conducts numerical experiments. Section 7 summarizes related work. Section 8 briefly concludes this paper with a summary of key points.

2. Virtual Network Embedding

This section defines the VN Embedding problem. This definition is conventional [5]–[9]. A PN is represented as an undirected graph $G_P = (V_P, E_P)$ consisting of physical nodes V_P and links E_P. Each physical node $v \in V_P$ is associated with available capacity $a(v) \in \mathbb{R}_{\geq 0}$ (CPU and/or memory), type $t(v) \in \mathbb{N}$ (functionalities and/or locations), and cost per unit capacity $c(v) \in \mathbb{R}_{\geq 0}$. We assume that required node resources do not depend on the number of virtual links connected to the node, because basic packet forwarding functionality is processed by dedicated forwarding engines or processors [18]. Each link $e \in E_p$ has available capacity $a(e) \in \mathbb{R}_{\geq 0}$ (bandwidth) and cost per unit capacity $c(e) \in \mathbb{R}_{\geq 0}$. Thus we write a PN as $P = (G_P, a, c, t)$.

A VN is an undirected *complete* graph, $G_V = (V_V, E_V)$ where $|E_V| = \binom{|V_V|}{2}$. Each virtual node $v \in V_V$ has capacity requirement $r(v) \in \mathbb{R}_{\geq 0}$ and type $t(v) \in \mathbb{N}$. Each virtual link $e \in E_V$ has capacity requirement $r(e) \in \mathbb{R}_{\geq 0}$. Hence, we write a VN as $N = (G_V, r, t)$. We assume a VN is a complete graph in this paper; note that we can readily convert VNs with non-complete topology into complete ones by adding zero capacity links.

An embedding is a pair of node mapping $M_N : V_V \to V_P$ and link mapping $M_L : E_V \times E_P \to [0, 1]$. Virtual node $v \in V_V$ is embedded in physical node $M_N(v) \in V_P$ that has enough capacity and matching type. A physical node can contain at most one virtual node per embedding. Virtual link $e \in E_V$ is embedded in physical *paths* and $M_L(e, f)$ is the proportion of embedding in physical link $f \in E_P$; the physical network supports path splitting [6] and $M_L(e, f)$ can be a fractional number in $[0, 1]$. A physical link can contain more than one virtual link.

In VNE, an InP takes a requested VN, N, and a PN, P, as input, and it finds the embedding, (M_N, M_L), that minimizes its embedding cost $c(M_N, M_L)$, which is the sum of all physical resource costs, as follows,

$$\begin{aligned} c(M_N, M_L) = &\sum_{v \in V_V} r(v) c\left(M_N(v)\right) \\ &+ \sum_{e \in E_V} \sum_{f \in E_P} r(e) c(f) M_L(e, f). \end{aligned} \tag{1}$$

Several VNE heuristics use LP [19] as a subroutine [3]. The time complexity of LP is $O(n^{3.5})$ with the interior point method [20], where n is the number of LP variables and $n = O(|E_V||E_P|)$ in VNE. The simplex method [19], which is also used to solve LP, is theoretically an exponential algorithm, but it is known to run in $O(n^3)$ in practice [21]. We assume LP can be solved in $O(n^3)$ in this paper.

3. Restriction Operations

This section introduces *restriction*, the new VNE operation class. Section 3.1 defines the operation. Section 3.2 describes two metrics for the operation, and Sect. 3.3 presents two instances of restriction operation.

3.1 Definition of Restriction

Restriction is a VNE operation class that converts a VN, N, into a restricted VN, N'. Restriction operations must satisfy the following three conditions:

A. The number of virtual links must be decreased, $|E(N)| > |E(N')|$, to lower the embedding time of $\left(M'_N, M'_L\right)$,

B. The feasible region can shrink, but never expand (that is, restricted); i.e., for any PN, if restricted N' is feasible, then the original N is also feasible, but the inverse does not necessarily hold, and,

C. Any feasible embedding for restricted N', $\left(M'_N, M'_L\right)$, can be converted into a corresponding feasible embedding of original N, (M_N, M_L), such that $c(M_N, M_L) = c\left(M'_N, M'_L\right)$, and this conversion takes $O(|V_V||V_P| + |E_V||E_P|)$ time.

To recap, the restriction allows us to embed N' with less time and have the same embedding quality as the original N. Note that, we use the restricted VN N' only during the VNE calculation. Upon completion of the computation, embedding $\left(M'_N, M'_L\right)$ of the restricted VN is converted back to original one (M_N, M_L) by leveraging Condition C and we embed the original VN N by using it.

3.2 Capacity Ratio and Feasibility Ratio

To measure how reasonably restricted N' is, this subsection introduces two metrics, *capacity ratio* and *feasibility ratio*.

3.2.1 Capacity Ratio

We first define the total required capacity of VN, N, as,

$$r(N) = \sum_{v \in V_V} r(v) + \sum_{e \in E_V} r(e), \tag{2}$$

where we assume that the node and link capacities are appropriately weighted to allow summation. Since restriction operations increase this total capacity, as shown in Fig. 1, we have $r'(N') \geq r(N)$ for a restriction operation. Hence, we use the capacity ratio, $r'(N')/r(N) \geq 1$, to capture the increase in required capacity. This metric does not directly represent the embedding cost since it has no connection with PNs, but it is reasonable to consider that the embedding cost is roughly correlated with the total required capacity; this assumption is confirmed by the experiments Sect. 6.2.

3.2.2 Feasibility Ratio

Since the restriction operations can shrink the feasible region by increasing required link capacities, some good solutions may be lost. The feasibility ratio represents the extra physical capacity needed to restore the original feasible region.

Let $\mathcal{F}(N; P)$ be the set of feasible solutions of N on P, and let $h : \mathcal{F}(N'; P) \to \mathcal{F}(N; P)$ be the conversion function from the restricted embedding to the original one. Then, Condition B of Sect. 3.1 is formalized as $h(\mathcal{F}(N'; P)) \subseteq \mathcal{F}(N; P)$, that is, the restricted feasible region lies within the original one. If the inverse inclusion holds, $h(\mathcal{F}(N'; P)) \supseteq \mathcal{F}(N; P)$, then the restricted optimal embedding is equal to the original optimal one due to Condition C. We relax this condition by introducing parameter $\alpha \geq 1$:

$$h\left(\mathcal{F}(N'; \alpha P)\right) \supseteq \mathcal{F}(N; P), \tag{3}$$

where αP is a PN whose available capacity a is multiplied by α, i.e., $\alpha P = (G_P, \alpha a, c, t)$. The smaller α is, the less extra physical capacity is needed to restore the feasibility region. Thus, we define the *feasibility ratio* as the minimum α value such that for *any* PN, the condition (3) holds.

3.3 Link Reduction and Node Insertion

This subsection introduces two restriction operations; *link reduction* and *node insertion*. Since they are defined on a VN, without PNs, they can be performed very quickly. Although there might be better restriction operations involving PNs, they would be as complex as VNE itself.

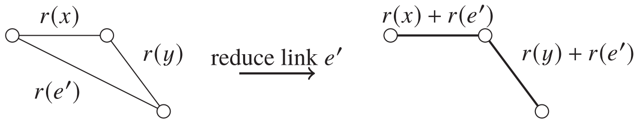

Fig. 1 Link reduction example. The required capacity of link e' is denoted by $r(e')$.

```
Algorithm 1 Link reduction
Input: A VN $(G_V, r, t)$ and a triangle $(e', x, y)$
Output: A restricted VN $(G'_V, r', t')$
1: $r(x) \leftarrow r(x) + r(e')$
2: $r(y) \leftarrow r(y) + r(e')$
3: $E_V \leftarrow E_V \setminus \{e'\}$
4: return $(G_V, r, t)$
```

3.3.1 Link Reduction

Let $e' \in E_V$ be a virtual link to be reduced. Link reduction collapses a triangle into a line on a VN, as shown in Fig. 1. The number of links is decreased by one, while the required capacities of the remaining links are increased by $r(e')$. This operation of reducing $e' \in E_V$ with respect to triangle (e', x, y), is formally described in Algorithm 1.

Here we describe the conversion from restricted embedding (M'_N, M'_L) to original embedding (M_N, M_L). If we reduce virtual link e' to triangle (e', x, y) and find restricted embedding (M'_N, M'_L), original embedding (M_N, M_L) is obtained as follows. Except for reduced virtual link e', restricted embedding (M'_N, M'_L) assigns all the virtual nodes and virtual links to physical nodes and physical paths, respectively. Hence we use those assignments as original embedding (M_N, M_L). We assign reduced link e' to the concatenation of the physical paths assigned to virtual link x and those assigned to virtual link y because virtual links x and y connect both endpoints of reduced link e'. The above conversion is mathematically expressed as follows:

$$M_N(v) = M'_N(v) \quad \forall v \in V_V, \tag{4}$$

$$M_L(e, f) = \begin{cases} M'_L(e, f) & \forall e \in E_V \setminus \{e'\}, \forall f \in E_P, \\ M'_L(x, f) + M'_L(y, f) & e = e', \forall f \in E_P. \end{cases} \tag{5}$$

The first two lines represent the assignments except for reduced link e', and the last line represents the assignments for reduced link e'. This conversion completes in $O(|V_V| + |E_V||E_P|)$ time. By these definitions, if restricted embedding (M'_N, M'_L) is feasible then original one (M_N, M_L) is also feasible, and they have the same cost; $c(M'_N, M'_L) = c(M_N, M_L)$. Thus, link reduction satisfies all the conditions of restriction given in Sect. 3.1.

Any sequence of link reduction is also considered as restriction; this is proved in Sect. 4.1. Therefore, we can

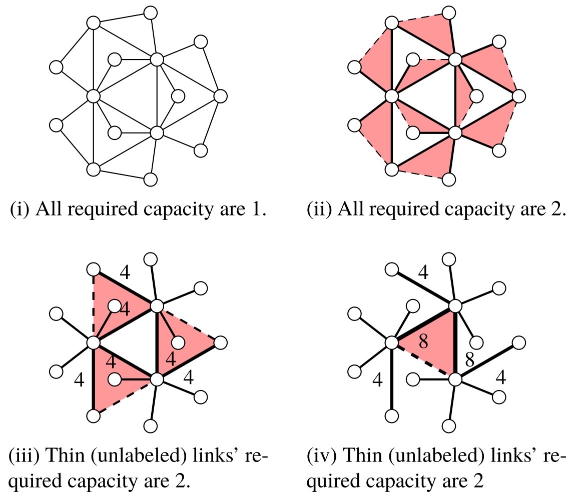
(i) All required capacity are 1.
(ii) All required capacity are 2.
4
4
4
4
4
4
8
8
4
4
(iii) Thin (unlabeled) links' required capacity are 2.
(iv) Thin (unlabeled) links' required capacity are 2

Fig. 2 Sequence of link reduction in which the maximum required capacity grows exponentially. The dashed link in red triangle is reduced.

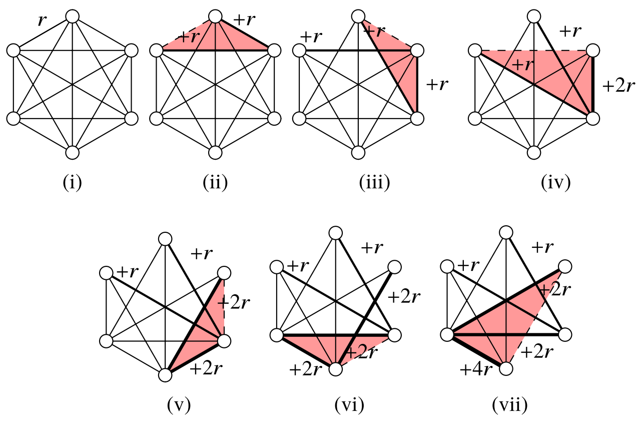

Fig. 3 Sequence of link reduction in which required capacity of a link r is added to remaining links multiple times; moreover, the multiplier increases at exponential rate. In each figure, the dashed link in red triangle is reduced. We omit the original capacities other than r.

reduce a VN as long as it has triangles, in order to decrease the VNE time as much as possible. However, multiple link reduction can increase the total required capacity at an exponential rate in the worst-case; Fig. 2 shows an example in which required capacity of one link grows exponentially, while Fig. 3 shows another example in which one link's required capacity r is repeatedly added. However, by applying link reduction in appropriate order, we can bound this ratio to polynomial form, as will be described in Sect. 5.1.

3.3.2 Node Insertion

Node insertion emulates traditional hierarchical network design. This operation adds core nodes to the VN, and connects original virtual nodes to one of the core nodes (Fig. 4). If the original nodes are partitioned into k groups ($k \leq |V_V|$) and k core nodes are inserted, this operation is called k-node insertion. k-node insertion is described in Algorithm 2; $E(X, Y)$ is a set of links that forms a complete bipartite graph between

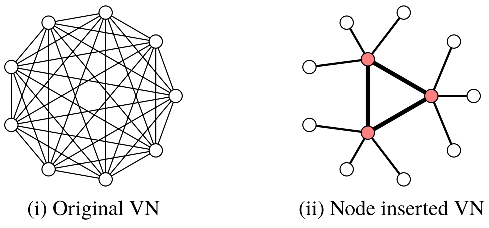

Fig. 4 An example of node insertion when the number of nodes $|V_V|$ and partitions k are 9 and 3, respectively. Newly added core nodes are solid red. The number of links in original $|E|$ is 36 (4i) and that of node inserted $|E'|$ is reduced to 12 (4ii).

Algorithm 2 Node insertion

```
Input: A VN $(G_V, r, t)$ and a node partition $\{V_1, \ldots, V_k\}$
Output: A restricted VN $(G'_V, r', t')$
 1: $V'_V \leftarrow V_V \cup \{u_1, u_2, \ldots, u_k\}$                ▷ add core nodes
 2: $E'_V \leftarrow \emptyset$
 3: for all pair of newly inserted nodes $u_i$ and $u_j$ do
 4:     $E'_V \leftarrow E'_V \cup \{u_i, u_j\}$                   ▷ connect core nodes
 5:     $r'(\{u_i, u_j\}) \leftarrow \sum_{e \in E(V_i, V_j)} r(e)$
 6: for $i \leftarrow 1, \ldots, k$ do
 7:     for all virtual node $v \in V_i$ do
 8:         $E'_V \leftarrow E'_V \cup \{\{u_i, v\}\}$          ▷ connect to the core node
 9:         $r'(\{u_i, v\}) \leftarrow \sum_{e \in E(\{v\}, V \setminus \{v\})} r(e)$
10: return $\left((V'_V, E'_V), r', t\right)$
```

the two node sets, X and Y; newly added node u_i has zero required capacity. New required link capacity r' is calculated as in line 5 and line 9 of Algorithm 2. The time complexity is $O(|E_V|)$.

k-node insertion is a restriction. We begin with condition A. The number of links in the node-inserted VN $|E(N')|$ is

$$|E(N')| = \sum_{i=1}^{k} |V_i| + \binom{k}{2} = |V_V| + \binom{k}{2}. \tag{6}$$

Since k is usually much smaller than $|V_V|$, $|E(N')|$ is smaller than the number of original links, $|E(N)| = \binom{|V_V|}{2}$.

Next, we consider Conditions B and C. Algorithm 3 transforms any node-inserted embedding, $\left(M'_N, M'_L\right)$, into an original embedding, (M_N, M_L). By their definitions, (M_N, M_L) is feasible and the cost, $c(M_N, M_L)$, is equal to the restricted one, $c\left(M'_N, M'_L\right)$.

4. Proposed Link Reduction Schemes

In this section, first, we show that any sequence of link reduction is a restriction operation (Sect. 4.1). Next, we propose a sequence of link reduction, *insert-emulated link reduction*, that can *emulate* the node insertion (Sect. 4.2). Then, we introduce two new sequences of link reduction: *minimum caption link reduction* and *star-based link reduction* (Sect. 4.3). The former greedily applies the link reduction, while the latter achieves the optimal capacity ratio among the link reduction schemes. In the next Sect. 5, we analyze these three link reduction schemes with respect to the capacity

Algorithm 3 Embedding conversion for node insertion

```
Input: An embedding of node-inserted VN $(M'_N, M'_L)$
Output: An embedding of original VN $(M_N, M_L)$
1: $(M_N, M_L) \leftarrow (M'_N, M'_L)$                              ▷ copy
2: for $i \leftarrow 1, 2, \ldots, k$ do
3:     for all pair of virtual nodes $v, w \in V_i$ do
4:         for all physical link $f \in E_P$ do
5:             $M_L(\{v, w\}, f) \leftarrow M'_L(\{v, u_i\}, f) + M'_L(\{u_i, w\}, f)$
6: for all pair of virtual nodes $x \in V_i,\ y \in V_j,\ i, j \in \{1, 2, \ldots, k\}$ do
7:     for all physical link $f \in E_P$ do
8:         $M_L(\{x, y\}, f) \leftarrow M'_L(\{x, u_i\}, f) + M'_L(\{u_i, u_j\}, f) + M'_L(\{u_j, y\}, f)$
9: return $(M_N, M_L)$
```

Table 1 Notations.

Symbols	Meanings
s	Number of link reductions
N_i	Restricted VN after i-th link reduction
V_i, E_i, r_i	Set of virtual nodes, links, and required capacity of N_i
$(M_{N,i}, M_{L,i})$	Embedding of N_i
$e_i, (e_i, x_i, y_i)$	Reduced link and triangle at i-th link reduction
S_i	Set of links that will be reduced at and after i-th link reduction, $\{e_i, e_{i+1}, \ldots, e_s\}$

ratio, feasibility ratio, and time complexity.

We use the notations of Table 1. We omit subscript V which represents *virtual* for clarity.

4.1 Sequence of Link Reduction

We argue that any sequence of link reduction operations is also a restriction. Condition A of restriction is trivial. We show constructive proof for Conditions B and C by using the following lemma.

Lemma 1. *For any $i \in \{1, \ldots s-1\}$, the following $(\tilde{M}_N, \tilde{M}_L)$ is a feasible embedding of N_i, and its cost, $c(\tilde{M}_N, \tilde{M}_L)$, equals the cost of the following link reduction, $c(M_{N,i+1}, M_{L,i+1})$,*

$$\tilde{M}_N(v) = M_{N,i+1}(v) \quad (\forall v \in V), \tag{7}$$

$$\tilde{M}_L(e, f) = M_{L,i+1}(e, f) \quad (\forall e \in E_{i+1}, f \in E_P), \tag{8}$$

$$\tilde{M}_L(e_i, f) = M_{L,i+1}(x_i, f) + M_{L,i+1}(y_i, f) \quad (\forall f \in E_P). \tag{9}$$

Proof. First, we prove feasibility. It is sufficient to consider the constraints with respect to reduced link e_i. We begin with topology constraints (virtual link must be embedded in physical paths), followed by capacity constraints. Topology constraints are satisfied because a triple of links, (e_i, x_i, y_i), forms a triangle. Capacity constraints also hold, since for any physical link $f \in E_P$, we have

$$\sum_{e \in E_i} r_i(e)\tilde{M}_L(e, f) \tag{10}$$

$$= \sum_{e \in E_i \setminus \{e_i, x_i, y_i\}} r_i(e)\tilde{M}_L(e, f) + r_{i+1}(x_i)M_{L,i+1}(x_i, f) + r_{i+1}(y_i)M_{L,i+1}(y_i, f) \tag{11}$$

$$= \sum_{e \in E_{i+1}} r_{i+1}(e)M_{L,i+1}(e, f) \leq a(f). \tag{12}$$

Equality of embedding cost is trivial by using (12) and the definition of embedding cost. □

Thus, by repeatedly applying Lemma 1, Algorithm 4 converts an embedding of a reduced network $(M_{N,s}, M_{L,s})$ to a feasible and equal-cost embedding of the original (M_N, M_L). Finally, this algorithm completes in $O(|V_V||V_P| + |E_V||E_P|)$ time.

Algorithm 4 Embedding conversion for link reduction

```
Input: An embedding of reduced network $N_s$: $(M_{N,s}, M_{L,s})$
Output: An embedding of original network $N$: $(M_N, M_L)$
1: $(M_N, M_L) \leftarrow (M_{N,s}, M_{L,s})$                        ▷ copy
2: for $i \leftarrow s, s-1, \ldots 1$ do
3:     for all physical link $f \in E_P$ do
4:         $M_L(e_i, f) \leftarrow M_L(x_i, f) + M_L(y_i, f)$
5: return $(M_N, M_L)$
```

4.2 Node Insertion Included by Link Reduction

We show that it is sufficient to consider link reduction rather than node insertion; for any node insertion operation, there exists a link reduction operation that produces a *better* restricted VN.

Theorem 1. *For any VN, N, and any k-node insertion, there exists a sequence of link reduction that satisfies the following:*

1. *Reduced network N_R is a subgraph of inserted network N_I.*
2. *Required link capacity of reduced network, r_R, is less than or equal to that of the corresponding inserted network, r_I.*

The first condition says that there exists an injective mapping, f, that maps nodes of reduced network $V(N_R)$ to nodes of inserted network $V(N_I)$, such that for any pair of nodes $u, v \in V(N_R)$ if $\{u, v\} \in E(N_R)$ then $\{f(u), f(v)\} \in E(N_I)$. The second condition says that for all links in N_R, the corresponding link in N_I has greater than or equal required capacity:

$$\forall \{u, v\} \in E(N_R) : r_R(\{u, v\}) \leq r_I(\{f(u), f(v)\}). \tag{13}$$

Theorem 1 only claims the existence of such a sequence, but we can implement an actual sequence as in the proof of Theorem 1. We called this restriction *insert-emulated link reduction* or Insert.

Finally, we give the proof of Theorem 1.

Proof of Theorem 1. Let $\{V_1, V_2, \ldots, V_k\}$ be a partition with respect to k-node insertion. We arbitrarily choose node w_i from each node set V_i and define a set of links E_W as follows:

$$E_W = \bigcup_{i=1}^{k} \{\{w_i, v\} \mid v \in V_i \setminus \{w_i\}\} \cup \left\{\{w_i, w_j\} \mid i \neq j\right\}. \tag{14}$$

For every pair of nodes $x, y \in V$, $d_W(x, y)$ denotes the number of links on the shortest x-y path in graph (V, E_W). We partition E_W by d_W:

$$E_i = \{\{x, y\} \mid d_W(x, y) = i\}. \tag{15}$$

Note that, for $i > 3$, the above set of links E_i is an empty set because the diameter of graph (V, E_W) is 3.

In the following, we construct a sequence of link reduction that satisfies the two conditions of Theorem 1; we begin with the subgraph condition followed by the capacity condition. The link reduction consists of two stages. In the first stage, we reduce $e \in E_3$ with respect to triangle (e, x, y) such that $x \in E_1$ and $y \in E_2$. There always exists such links, x, y, due to the definition of E_i. Hence, we can complete link reduction on all links in E_3. In the second stage, we reduce $e \in E_2$ with respect to triangle (e, x, y) such that $x, y \in E_1$. There always exists such links for the same reason as E_3. The topology of the reduced network is a subgraph of the inserted one, because the set of links in the reduced network equals $E_1 = E_W$.

Next, we discuss the capacity condition. There remains arbitrariness of the above two staged link reduction: the reduction order of links within E_3 and E_2 and the choice of triangles. In the rest of this subsection, we show that the following equalities hold regardless of the choices made:

$$r_R(\{v, w_i\}) = \sum_{e \in E(\{v\}, V \setminus \{v\})} r(e) \quad (\forall v \in V_i), \tag{16}$$

$$r_R\left(\{w_i, w_j\}\right) = \sum_{e \in E(V_i, V_j)} r(e) \quad (i \neq j). \tag{17}$$

As described above, every link in E_3 is reduced with respect to a triangle that includes links of E_1 and E_2. Similarly, every link in E_2 is reduced with respect to a triangle with two E_1's links. Hence, after link reduction, the required capacity of E_3 link is added to three E_1 links, and that of E_2 link is added to two E_1 links. These E_1 links are the shortest paths between the end points of reduced links and the shortest path in graph (V, E_W) is unique for each pair of nodes, and so is independent from reduction order and choice of triangles. Thus, each required capacity is added on the shortest path. This yields (16) and (17). □

Note that the number of links $|E|$ is now reduced to $|E'| = |E_W| = |V| - k + \binom{k}{2}$.

4.3 Link Reduction Schemes

We propose two link reduction schemes in addition to the inserted-emulated link reduction introduced in Sect. 4.2. The first scheme, *minimum capacity link reduction* or Min, greedily applies the link reduction and experientially produces better embedding than Insert, as we will show in Sect. 6. The second scheme, *star-based link reduction* or Star, has analytically better capacity ratio, feasibility ratio, and time complexity than the other two schemes. It achieves the theoretical optimal capacity ratio among the link reduction schemes (Sect. 5.1.3).

Algorithm 5 Minimum capacity link reduction

```
Input: A VN $(G_V, r, t)$ and a maximum number of reduced links $s$
Output: A restricted VN $(G'_V, r', t')$ that has at least $|E| - s$ links
1: Initialize counter: $i \leftarrow 0$
2: while There exists a triangle and $i < s$ do
3:     $e \leftarrow$ Find a minimum capacity link that belongs to a triangle
4:     $(G'_V, r', t') \leftarrow$ Reduce the link $e$ to a triangle
5:     Increment counter: $i \leftarrow i + 1$
6: return $(G'_V, r', t')$
```

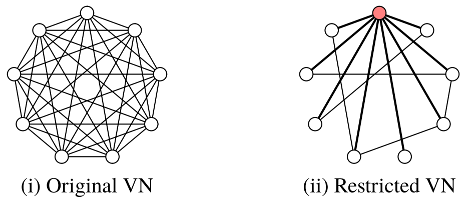

Fig. 5 An example of star-based reduction when the number of virtual nodes $|V|$ and reduced links s are 9 and 24, respectively. The center of the star is solid red. The number of links $|E|$ in original (5i) is 36 and that of restricted (5ii) $|E'|$ is reduced to $12 = 36 - 24$.

4.3.1 Minimum Capacity Link Reduction (Min)

This scheme repeatedly reduces minimum capacity link a given number of times. Algorithm 5 describes this scheme. At every step, this algorithm finds a minimum capacity link e that belongs to a triangle and reduces it. If the network does not have any triangle, then the algorithm outputs the current network and halts. Hence, Min does not always applies link reduction the given number of times s. While, the other scheme Star always does.

4.3.2 Star-Based Link Reduction (Star)

This restriction converts a VN into a star with extra links (Fig. 5) as described in Algorithm 6. The algorithm first selects a center node, v^*, so that node v^* maximizes the connected link capacity $r(E(v)) = \sum_{e \in E(v)} r(e)$ where the set of links that connects to node v is denoted by $E(v)$. Then, the algorithm finds the first s smallest capacity links that do not connect to the center node v^*. We write these links as $\{e_1, e_2, \ldots, e_s\}$. Next, for each link $e \in \{e_1, \ldots, e_s\}$, the algorithm reduces the selected link e to the triangle (e, v^*). This removes s links in total from the original VN.

Note that this algorithm works correctly, i.e., at every link reduction of link e, the network has the other two links that form the triangle (e, v^*). This is because, the two links

Table 2 Link reduction schemes and metric upper bounds.

Scheme	Capacity ratio	Feasibility ratio	Feasibility ratio with bounded $q = \frac{\max r(e)}{\min r(e)}$	Time complexity
Min	$1 + s$	$2^{\Theta(V)}$	$1 + s^2 q$	$O\left(\lvert V\rvert^3\right)$
Insert	3	∞	$\lvert V\rvert^2 q/4$	$O\left(\lvert V\rvert^4\right)$
Star	$1 + \frac{2s}{\lvert V\rvert(\lvert V\rvert-1)}$	∞	$1 + \min(s, \lvert V\rvert - 1)q$	$O\left(\lvert V\rvert^2 \log \lvert V\rvert\right)$

Algorithm 6 Start-based reduction

```
Input: A VN $(G_V, r, t)$ and a number of reduced links $s$
Output: A restricted VN $(G'_V, r', t')$ that has $|E| - s$ links
1: $v^* \leftarrow \mathrm{argmax}\ \{r(E(v)) \mid v \in V_V\}$    ▷ find the center of a star
2: $\{e_1, e_2, \ldots, e_s\} \leftarrow$ Find first $s$ smallest capacity links that are not
   connected to center $v^*$
3: for all link $e \in \{e_1, e_2, \ldots, e_s\}$ do
4:     Reduce link $e$ to triangle $(e, v^*)$
5: return $(G_V, r', t)$
```

are connected to center v^* and the algorithm only reduces the links that do not connect to center v^*.

5. Analysis of Link Reduction Schemes

In this section, we analyze the three link reduction schemes, Min, Insert, and Star, using the metrics of the capacity ratio (Sect. 5.1), feasibility ratio (Sect. 5.2), and time complexity (Sect. 5.3). We summarize the results in Table 2. In this table, the maximum ratio between required link capacity r is denoted by q. That is, q equals $\max_{e\in E} r(e)/\min_{f\in E} r(f)$. Section 6 empirically analyzes the performance of the three schemes.

5.1 Capacity Ratio

We evaluate the capacity ratio of the three link reduction schemes introduced in Sect. 4.2, Sect. 4.3; Min, Insert, and Star (the second column in Table 2).

5.1.1 Min

Section 3.3.1 shows that repeated link reduction can increase the required link capacity exponentially (Figs. 3 and 2), but we will show that the capacity ratio is bounded by the number of link reductions s.

Theorem 2. *Capacity ratio does not exceed $s + 1$ as long as we reduce the minimum capacity link at every step:*

$$\frac{r'(N')}{r(N)} \leq s + 1 \tag{18}$$

Proof. At the i-th step, link reduction increases the required capacity of only two links by $r_i(e_i)$, and reduced link e_i is removed. Thus, we can bound the total required capacity of S_{i+1} as follows:

$$r_{i+1}(S_{i+1}) \leq r_i(S_i) + r_i(e_i). \tag{19}$$

Due to the minimality of required capacity $r_i(e_i)$, we have

$$r_i(e_i) \leq \frac{r_i(S_i)}{|S_i|} = \frac{r_i(S_i)}{s - (i-1)}. \tag{20}$$

By repeatedly applying these equations, we have

$$\begin{aligned} r_i(e_i) &\leq \frac{1}{s-(i-1)} \cdot r_i(S_i) && (21)\\ &\leq \frac{1}{s-(i-1)} \cdot \frac{s-(i-2)+1}{s-(i-2)} \cdot r_{i-1}(S_{i-1}) && (22)\\ &\cdots \\ &\leq \frac{1}{s-(i-1)} \frac{s-(i-2)+1}{s-(i-2)} \cdots \frac{s}{s-1} \cdot \frac{s+1}{s} r_1(S_1) && (23)\\ &\leq \frac{s+1}{(s-(i-1))\,(s-(i-2))} r_1(S_1). && (24) \end{aligned}$$

A link reduction increases the total required capacity, $r_i(N_i)$, by $r_i(e_i)$. Hence we can bound the capacity ratio, as follows:

$$\begin{aligned} \frac{r'(N')}{r(N)} &= 1 + \frac{1}{r(N)} \sum_{i=1}^{s} r_i(e_i) && (25)\\ &\leq 1 + \frac{r_1(S_1)}{r(N)} \left\{ \sum_{i=1}^{s} \frac{s+1}{(s-(i-1))(s-(i-2))} \right\} && (26)\\ &\leq 1 + s && (27) \end{aligned}$$

□

Note that if we replace (19) with the following inequation,

$$r_{i+1}(S_{i+1}) \leq r_i(S_i), \tag{28}$$

the bound is greatly reduced to $2 + \log s$ from $1 + s$. In fact, (19) is not tight enough, because $r_i(S_i)$ has the following three cases:

$$r_{i+1}(S_{i+1}) - r_i(S_i) = \begin{cases} -r_i(e_i) & x_i, y_i \notin S_i \\ +r_i(e_i) & x_i, y_i \in S_i \\ 0 & \text{otherwise.} \end{cases} \tag{29}$$

In later steps of reduction, (19) is likely to be a loose bound; $r_i(S_i)$ is more likely to decrease than increase. This is because the total capacity of S_i, $r_{s+1}(S_{s+1})$, must be 0 after all link reductions. Thus, we can conjecture that there will be a much smaller upper bound; supplemental experiments (Sect. 6.2) support our conjecture.

Here, we analyze the relationship between total capacity growth $\Delta R = |r'(N') - r(N)|/r(N)$ and computation time speed-up $\Delta T = |T' - T|/T'$, under our conjecture $r'(N')/r(N) \leq 2 + \log s$, where T' and $T = \Theta(E^3)$ are computation times with and without minimum capacity link reduction, respectively. Finally, the speed-up is given, as follows:

$$\Delta T \geq \left(\frac{1}{1 - \exp(\Delta R - 1)/|E|}\right)^3 - 1. \tag{30}$$

The exponential speed-up is induced by the linear increase in required capacity.

5.1.2 Insert

The topology and required capacity of Insert were given in Sect. 4.2. Hence, it is easy to analyze and offers a much better upper bound than the Min scheme.

Theorem 3. *For any partition of virtual nodes, the capacity ratio of Insert does not exceed* 3.

Proof. From (16) and (17), the total capacity ratio of restricted VN, $r'(N')$, is bounded as follows:

$$\begin{aligned} r'(N') &= \sum_{v \in V} r(v) + \sum_{v \in V} \sum_{e \in E(\{v\}, V \setminus \{v\})} r(e) \\ &\quad + \sum_{i \neq j} \sum_{e \in E(V_i, V_j)} r(e) && (31) \\ &\leq 2r(N) + \sum_{i \neq j} \sum_{e \in E(V_i, V_j)} r(e) && (32) \\ &\leq 3r(N) && (33) \end{aligned}$$

□

The last term of (32), $\sum_{i \neq j} \sum_{e \in E(V_i, V_j)} r(e)$, is the cut value of the partition, $\{V_1, V_2, \ldots, V_k\}$, and this is the only part that depends on the partition. Thus, to reduce total capacity $r'(N')$, we need to reduce the k-cut value.

5.1.3 Star

We show that Star has the *smallest* capacity ratio upper bound among the three schemes.

Theorem 4. *The capacity ratio of Star reduction is at most*

$$1 + \frac{2s}{|V|(|V| - 1)}. \tag{34}$$

Proof. Let v^* be the center of the star, i.e., the node maximizes the sum of connected link capacity $r(E(v)) = \sum_{e \in E(v)} r(e)$. Let $S = \{e_1, e_2, \ldots, e_s\}$ be the set of reduced links.

From the maximality of center node v^*, we have

$$2 \cdot r(E) = \sum_{v \in V} r(E(v)) \leq |V| \cdot \sum_{e \in E(v^*)} r(e). \tag{35}$$

We also have

$$\frac{\sum_{e \in S} r(e)}{|S|} \leq \frac{\sum_{e \in E \setminus E(v^*)} r(e)}{|E \setminus E(v^*)|}, \tag{36}$$

this is because reduced links S are the first s smallest links in $E \setminus E(v^*)$. Whenever link $e \in S$ is reduced, the total capacity increases by exactly $r(e)$. Hence, the total capacity ratio is bounded as follows

$$\begin{aligned} r'(N') &= r(N) + \sum_{e \in S} r(e) && (37) \\ &\leq r(N) + \frac{|S|}{|E \setminus E(v^*)|}\left(r(E) - \sum_{e \in E(V^*)} r(e)\right) && (38) \\ &\leq r(N) + \frac{|S|}{|E \setminus E(v^*)|}\left(r(E) - \frac{2}{|V|} r(E)\right) && (39) \\ &\leq \left(1 + \frac{2s}{|V|(|V| - 1)}\right) r(N). && (40) \end{aligned}$$

□

This theorem says that the capacity ratio of Star never exceeds 2. While, we can easily construct a VN whose capacity ratio of Insert exceeds 2 and supplemental experiments in Sect. 6.2 shows that there exist a VN whose capacity ratio of Min exceeds 2.

Next, we prove that this upper bound is optimal, i.e., no reduction scheme has smaller capacity ratio upper bound than Star. That is, there is no link reduction scheme that makes capacity ratio less than $1 + \frac{2s}{|V|(|V|-1)}$ for *all* VNs.

Theorem 5. *There exists a VN N such that, for any sequence of link reduction that decreases the number of links by s, its capacity ratio is at least*

$$1 + \frac{2s}{|V|\,(|V| - 1)}. \tag{41}$$

Proof. Let N be a VN such that all links have the same capacity, say 1, and all nodes have zero capacity. Then, whenever a link is reduced, the total capacity ratio increases by at least 1. Hence, after reducing s links, the total capacity ratio increases at least s. Thus, we have the following capacity ratio evaluation

$$\frac{r'(N')}{r(N)} \geq \frac{|E| + s}{|E|} = 1 + \frac{2s}{|V|\,(|V| - 1)}. \tag{42}$$

□

This theorem says that if we are given a constant VN, that is, hard to minimize the capacity ratio, then Star successfully minimizes the capacity ratio.

Finally, we show that Star achieves the minimum capacity ratio not only for the constant instance but also for a wider class of instances. We show that Star minimizes capacity ratio when the star is simultaneously a maximum spanning tree with respect to capacity r.

Theorem 6. *If star $E(v^*)$ is a maximum spanning tree of the graph (V, E, r), then* Star *minimizes the capacity ratio among the reduction schemes that decreases the number of links by s.*

Proof. Proof by contradiction. Suppose that there exists a sequence of link reductions σ that achieves smaller total capacity than Star. Let $N_\sigma = ((V, E_\sigma), r_\sigma, r)$ be the restricted VN yielded by the link reduction σ. Let $S = \{e_1, e_2, \dots, e_s\}$ be the set of reduced links in Star.

Then, in Star, every link is reduced to the center node and total capacity $r'(N')$ is

$$r'(N') = r(N) + \sum_{e \in S} r(e) = r(N) + r(E) - r(E \setminus S). \tag{43}$$

Also, in link reduction σ, whenever link e is reduced, the total capacity increases by at least $r(e)$, so we have

$$r_\sigma(N_\sigma) \geq r(N) + \sum_{e \in E \setminus E_\sigma} r(e) = r(N) + r(E) - r(E_\sigma). \tag{44}$$

Because star $E(v^*)$ is a maximum spanning tree and reduced links S are the first s smallest links among $E \setminus E(v^*)$, we have the following

$$r(E \setminus S) \geq r(E_\sigma). \tag{45}$$

Together with these three equations, (43), (44), and (45), we have $r'(N') \leq r_\sigma(N_\sigma)$. However, this contradicts to the assumption that link reduction σ has smaller total capacity than Star. □

5.2 Feasibility Ratio

We evaluate the feasibility ratio of the three schemes: Min, Insert, and Star. However, exact evaluation of feasibility ratio is difficult because it requires investigation of all PNs and check of the feasibility by scaling available capacity. Hence, we, first show two theorems (Theorems 7 and 8) that give us upper and lower bounds of feasibility ratio without inspecting any PN. We then, by using the theorems, show the feasibility ratio of each scheme, third and fourth columns in Table 2.

5.2.1 Upper and Lower Bounds

We can evaluate the feasibility ratio from above by using the change in link capacity. Also, we can evaluate the feasibility ratio from below by using the change in the connected capacity of nodes. In the rest of this subsection, we use these two theorems to evaluate the feasibility ratios of Min, Insert, and Star schemes.

Theorem 7. *Let γ be the relative increase in link capacity caused by a restriction, i.e.,*

$$\gamma = \max_{e \in E'} \frac{r'(e)}{r(e)}. \tag{46}$$

Then, the feasibility ratio of the restriction is at most γ.

Theorem 8. *Let $k < |V|$ be a positive integer, and δ be the relative increase in the minimum connected capacity of among node sets with size k, i.e.,*

$$\delta = \frac{\min_{|U'|=k} r'(E'(U'))}{\min_{|U|=k} r(E(U))} \tag{47}$$

where $E(U)$ is a set of links between the two set of nodes U and $V \setminus U$ in the original VN. Then, the feasibility ratio is at least δ.

Proof of Theorem 7. Let P be a PN such that there exists a feasible embedding, (M_N, M_L), between VN N and PN P. Then, the following embedding, $\left(M'_N, M'_L\right)$, is a feasible embedding between restricted VN N' and scaled PN γP:

$$M'_N(v) = M_N(v) \quad (\forall v \in V'), \tag{48}$$
$$M'_L(e, f) = M_L(e, f) \quad (\forall e \in E', f \in E_P). \tag{49}$$

Hence, the feasibility ratio is at most γ. □

Proof of Theorem 8. Let P be a PN that has the same topology and capacity as VN N. Then, the embedding between VN N and PN P is feasible because the trivial embedding is feasible. Here, trivial embedding embeds each virtual node onto the corresponding physical node.

Next, we consider an embedding between restricted VN N' and PN P. Any set of physical nodes, $W \subseteq V_P$ with size k, contains exactly k virtual nodes since VN and PN have the same number of nodes. We denote these nodes as $W' \subseteq V'$. These nodes, W', have connected $r'(E'(W'))$ link capacity in total where $E'(W')$ is the set of links between W' and $V' \setminus W'$ in restricted VN N'. While the corresponding physical nodes W have connected $r(E(W))$ link capacity in total. Thus, in order to be feasible, the PN must be scaled at least $r'(E'(W'))/r(E(W)) \geq \min_{|U'|=k} r'(E'(U'))/r(E(W))$. This argument holds for any set of nodes W with size k. Hence, in order to be feasible, the PN must be scaled at least δ. □

5.2.2 Min

The trivial upper bound of the feasibility ratio of Min is $2^{|E|}$. This is because, at every link reduction, the capacity increase of a link is bounded by two, $r'(e) \leq 2r(e)$, due to the minimality of reduced link capacity. This upper bound can be improved, as follows (proof in Appendix A.1):

Corollary 9. *The feasibility ratio of* Min *is $2^{\Theta(|V|)}$.*

However, if the ratio between maximum and minimum required link capacities, $\max_{e \in E} r(e)/\min_{f \in E} r(f)$, is bounded above by a constant, then the upper bound becomes very small.

Corollary 10. *If* $\max_{e\in E} r(e)/\min_{e\in E} r(e) < q$*, then the feasibility ratio of* MIN *is bounded above by* $1 + s^2 q$.

Proof. We use Theorem 7 to obtain the upper bound. Hence, we consider the link capacity in the restricted VN. For any link $e \in E'$ in the restricted VN, its capacity is bounded above as follows due to the inequality (24):

$$r'(e) \leq r(e) + \sum_{i=1}^{s} r_i(e_i) \tag{50}$$

$$\leq r(e) + \sum_{i=1}^{s} \frac{s+1}{i(i+1)} r_1(S_1) = r(e) + s \cdot r_1(S_1). \tag{51}$$

Also, the sum of the link capacity over S_1 does not exceed $|S_1| \max_{f\in E} r(f)$. Thus, the relative increase of the link capacity $r'(e)/r(e)$ is bounded above by $1 + s|S_1| \max_{f\in E} r(f)/r(e) \leq 1 + s^2 q$. □

Note that, if we replace (19) with (28), then the feasibility ratio is bounded by $1 + s(1 + \log s)q$ as with Theorem 2.

5.2.3 INSERT

Naive INSERT may have unbounded feasibility ratio. Figure 6 shows an example of INSERT. Applying Theorem 8 with $k = 1$ to these VNs, confirms that the lower bound of the feasibility ratio is at least $1/3\epsilon$. This is because, in the original VN, node v_2 has the minimum connected link capacity 3ϵ. While, in the restricted VN, node v_1 has the minimum connected link capacity $2+\epsilon$. Since ϵ can be minimized arbitrarily, the feasibility ratio cannot be bounded.

We here discuss two strategies that keep the feasibility ratio small: utilize a partition of minimum k-cut $(\{\{v_2\}, V \setminus \{v_2\}\}$ in Fig. 6), or select core nodes of maximum link capacities, $\sum_{e\in E(\{v\}, V\setminus\{v\})} r(e)$ (v_1 becomes core in Fig. 6). These strategies, however, do not work well in general, and the feasibility ratio still cannot be bounded. Finally, we have the following (proof in Appendix A.2).

Corollary 11. *For any real number* $\epsilon > 0$*, there exists a VN and a PN whose feasibility ratio of* INSERT *is at least* $1/\epsilon$ *even if the node partition is min k-cut and core nodes have the maximum connected link capacities.*

As in the MIN scheme case, if the ratios between original link capacities are bounded above by a constant, then the upper bound is reduced to polynomial.

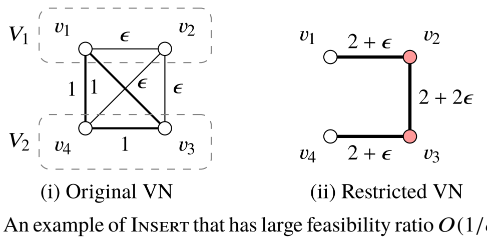

Fig. 6 An example of INSERT that has large feasibility ratio $O(1/\epsilon)$. Core nodes are filled in red.

Corollary 12. *If* $\max_{e\in E} r(e)/\min_{e\in E} r(e) < q$*, then the feasibility ratio of* INSERT *is bounded above by* $|V|^2 q/4$.

Proof. We use Theorem 7 to obtain the upper bound. Hence, we consider the link capacity in the restricted VN. If link $e \in E'$ in the restricted VN connects two core nodes, then its capacity is bounded above by $|V|^2 \max_{e\in E} r(e)/4$ due to (17). Otherwise, the link capacity is bounded above by $(|V|-1) \max_{e\in E} r(e)$ due to (16). Thus, the relative increase in link capacity is bounded by $|V|^2 q/4$. □

5.2.4 STAR

This scheme also has, in general, unbounded feasibility ratio similar to INSERT (proof in Appendix A.3).

Corollary 13. *For any real number* $\epsilon > 0$*, there exists a VN and a PN for which the feasibility ratio of* STAR *is at lest* $1/\epsilon$.

However, if the ratios between original link capacity are bounded above by a constant, the feasibility ratio is reduced to polynomial.

Corollary 14. *If* $\max_{e\in E} r(e)/\min_{e\in E} r(e) < q$*, then the feasibility ratio of* STAR *is bounded above by* $1 + \min(s, |V| - 2)q$.

Proof. We use Theorem 7 to obtain the upper bound. Thus, we consider the relative increase in link capacity. Let v^* be the center node of the star. Then, due to the construction of Algorithm 6, links that do not connect to center v^* retain their original capacity during the algorithm. This is because any link e that will be reduced is not connected to center node v^* but will be reduced to the triangle (e, v^*). Also, for any link e, the number of reduced triangles contained link e is at most $\min(s, |V| - 2)$. At every step, the capacity increases at most $\max_{e\in E} r(e)$. Hence, for any link $e \in E'$ in the restricted VN, the relative increase in link capacity is bounded above by $1 + \min(s, |V| - 2)q$. □

5.3 Time Complexity

This subsection describes the time complexity of the three schemes: MIN, INSERT, and STAR as in the last column of Table 2.

5.3.1 MIN

If minimum capacity link reduction is repeatedly performed as much as possible, the time complexity is $O(|E||V|)$. We sort links, E, in ascending order of required capacity $r(e)$ and store them in a priority queue. This takes $O(|E| \log |E|)$. At every step, we pop a minimum required capacity link e and check whether there is a triangle that contains e. If there is no such triangle, we pop the next link. If there are triangles, we reduce link e to one of them and update both required capacity and priority of remaining links. This single step takes $O(|V|)$ since priority updates occur only twice.

5.3.2 Insert

Given the node partition and core nodes, the time complexity is $O(|E|) = O\left(|V|^2\right)$ in Insert. This is because each link is referenced at most three times in (16) and (17). Computing a minimum k-cut is known to be $\mathcal{NP}$-hard. However, when k is an input, it can be solved in $O\left(|V|^{k^2}\right)$ [22]. There is an approximation algorithm whose approximation ratio is $(2-2/k)$ and computation cost is $O\left(|V|^4\right)$ [23].

5.3.3 Star

The time complexity of Star is $O\left(|V|^2 \log |V|\right)$. This is because, finding center node v^*, enumerating first s smallest links, and reducing these links takes $O(|V|)$, $O\left(|V|^2 \log |V|\right)$, and $O\left(|V|^2\right)$, respectively.

6. Experiments

In the experiments, we evaluate four methods; LP Relax [9] (*Normal*), LP Relax after Min link reduction, LP Relax after Insert link reduction, and LP Relax after Star link reduction. We compare the four methods in terms of computation time, evaluate their embedding costs, and show that link reduction speeds up the computation exponentially with only a linear increase in embedding cost. Then, we examine the trade-off between computation time and embedding cost and show that Min has the best trade-off among the three schemes. In Min, if there exist multiple triangles that contain the minimum capacity link, we choose the triangle that maximizes the sum of required link capacity[†]. In Insert, we partition virtual nodes by minimum k-cut approximately, as described in Sect. 5.3.2, and choose core nodes that have maximum connected link capacities.

Parameters used in the experiments are chosen following [5] (Table 3). Virtual network topologies are complete, and physical ones are random but connected. We pick physical network topologies uniformly at random from the set of connected graphs that have the given number of nodes and links. Available capacity a, cost c, node type t, and required node capacity r are random variables with uniform distributions. We compare four distributions on required link capacity r: constant $1/2$, uniform random distribution between 0 and 1, exponential distribution with parameter $1/2$, and lognormal distribution converted from standard normal distribution. Parameter β, *reduce ratio*, indicates how many times link reduction is performed; In Min, we apply link reduction up to $\beta|E_V|$ times as long as the VN has triangles. In Insert, we choose the number of partitions, k, as the minimum integer such that $|E'_V| \geq (1-\beta)|E_V|$. In Star, we reduce links $\beta|E_V|$ times. We solved 100 VNE instances for each parameter set, and so solved 26,000 VNE in total. Embedding success rates of Normal, Min, Insert, and Star were 99.90%, 99.95%, 96.11%, and 97.62%, respectively.

[†]Triangle selection slightly impacted both computation time and embedding cost in a preliminarily experiment that examined three types of triangle selection scheme: maximum link sum, minimum link sum, and random; maximum had the best performance.

Table 3 Parameters in experiments.

Parameter	Values
No. of virtual nodes $\lvert V_V\rvert$	$\{5, 10, 15, 20, 25\}$
No. of virtual links $\lvert E_V\rvert$	$\binom{\lvert V_V\rvert}{2}$
No. of physical nodes and links; $\lvert V_P\rvert$, $\lvert E_P\rvert$	100, 316
No. of node types	10
Required capacity of node; $r(v)$	[0, 1]
Cost of physical node and link; $c(v)$, $c(e)$	[0, 1]
Available capacity of node and link; $a(v)$	2, 100
Reduce ratio; β	$\{0.2, 0.4, 0.6, 0.8\}$

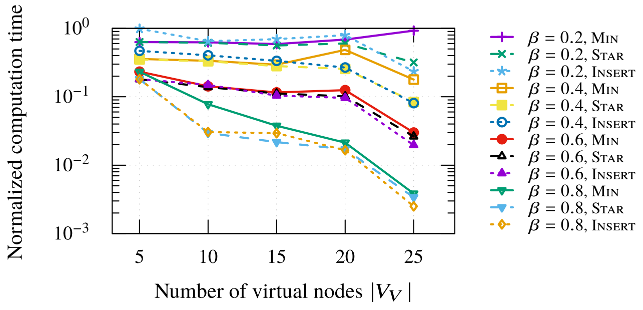

Fig. 7 Average normalized computation time versus number of virtual nodes when the capacity distribution is uniform. Solid, dashed, and dotted lines represent Min, Star, and Insert, respectively.

The experiments were conducted on a Linux machine with Intel Xeon E5–2640 v3 2.60 GHz. Our method was implemented in C++ and Haskell with GLPK [24] as an LP solver.

6.1 Computation Time

Link reduction greatly reduces the computation time. Figure 7 shows the average computation time and the number of virtual nodes $|V_V|$ when the capacity distribution is uniform. The computation time is normalized by that of Normal. Link reduction scheme slightly impact computation time, the major factors are the number of virtual nodes $|V_V|$ and the reduce ratio β. Figure 8 shows computation time of the other three distribution: constant, exponential, and lognormal. In these figures, we omit the time of Insert and Star for the sake of visibility because all the three reduction schemes took almost the same time as in Fig. 7. Also, in Fig. 8, computation time mainly depends on the VN size, i.e., the number of virtual nodes $|V_V|$ and reduce ratio β.

Over 99.9% of the time is consumed by solving LP in the three schemes. In fact, even when the number of virtual nodes $|V_V|$ is 25, the average time consumed by Min, Insert, and Star are less than 0.3, 0.9, and 0.002 seconds, respectively. As expected in the last column in Table 2, Star has the smallest computation time and Insert has the largest one. This is because the former only sorts links E_V with respect to the required capacity r, while the latter executes

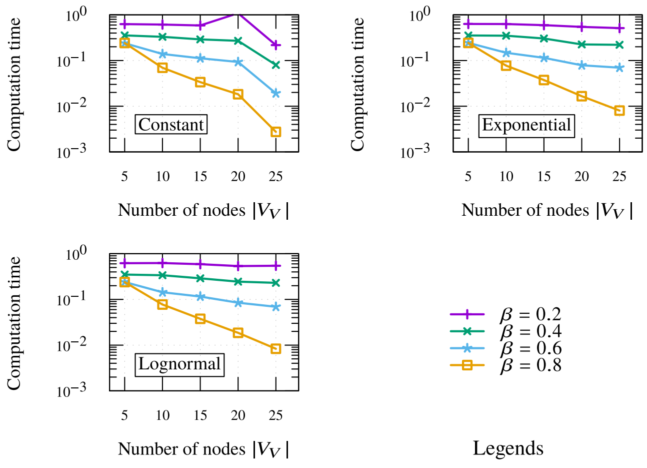

Fig. 8 Average normalized computation time of Min versus number of virtual nodes for constant, exponential, and lognormal capacity distribution.

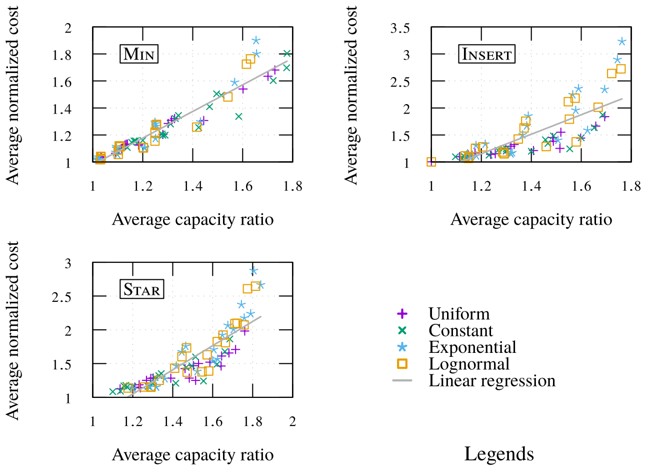

Fig. 9 Strong positive correlation between embedding cost and capacity ratio.

minimum k-cut approximation algorithm.

Link capacity distribution and reduction method slightly impact the computation time, while the major factors are the number virtual of nodes $|V_V|$ and the reduce ratio β. When the reduce ratio β is 0.4, the computation time is halved in all three schemes and the four link capacity distributions. When β is 0.8, it is reduced by three orders of magnitude. Note that this time reduction is explained as LP having roughly cubic time complexity and $(1-0.8)^3 = 0.008$ since link reduction reduces the number of LP variables to $O\left((1-\beta)|V_V|^2|E_P|\right)$ from $O\left(|V_V|^2|E_P|\right)$.

6.2 Embedding Cost

First, we draw scatter plots between embedding cost and capacity ratio (Fig. 9). The embedding cost is normalized by that of Normal like the computation time. Each point corresponds to a combination of three parameter values, the number of virtual nodes $|V_V|$, the reduce ratio β, and the capacity distribution. These points represent average normalized cost and capacity ratio over 100 VNE instances. As described in Sect. 3.2.1, we found a strong positive correlation between them: the correlation coefficients of Min, Insert, and Star are 0.95, 0.80, and 0.87, respectively. The results support our contention that the capacity ratio can be used as a measure of embedding cost.

Figure 10 shows the average embedding cost and the number of virtual nodes $|V_V|$ when the capacity distribution is uniform. The cost is normalized by Normal. Figure 11 shows embedding cost of the other three distributions. In these figures, we omit the cost of Insert and Star for the sake of visibility. The other two schemes, Insert and Star, has roughly the same tendency, but the Min had smaller embedding cost the others. We quantitatively compare the three schemes in Sect. 6.3.

Surprisingly, the average normalized costs of Min were less than two for all plots. The embedding cost increased sub-linearly with the number of removed links (i.e., much less than $O\left(|V_V|^2\right)$), as expected, $r'(N')/r(N) \leq 2+s$, in

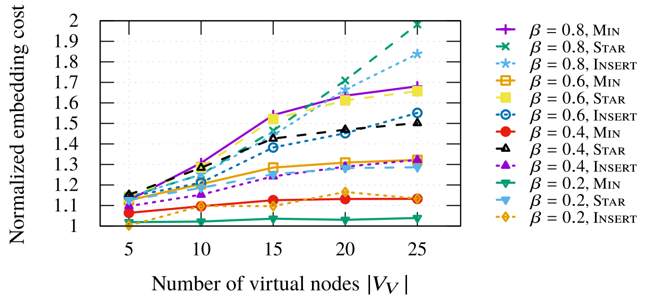

Fig. 10 Average normalized embedding cost versus number of virtual nodes when the capacity distribution is uniform. Solid, dashed, and dotted lines represent Min, Star, and Insert, respectively.

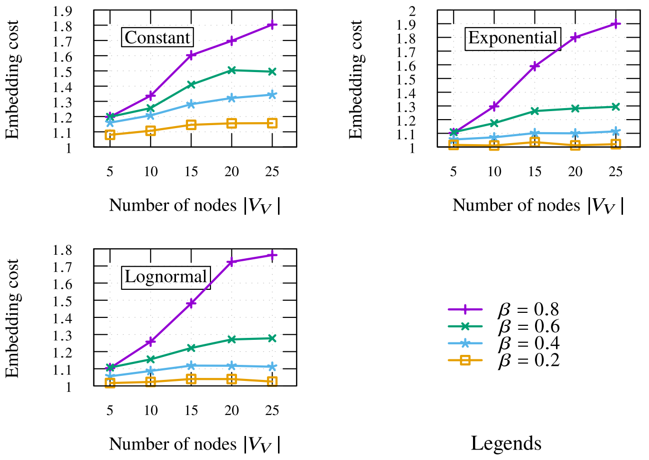

Fig. 11 Average normalized embedding cost of Min versus number of virtual nodes for each constant, exponential, and lognormal capacity distribution.

Sect. 5.1-2. Capacity distribution has some impact on the cost, but it is no significant.

In order to well understand the capacity ratio of Min, we conducted a supplemental numerical experiment; we measured the maximum capacity ratio for 1000 instances of each VN size. Figure 12 shows the maximum ratio. We found

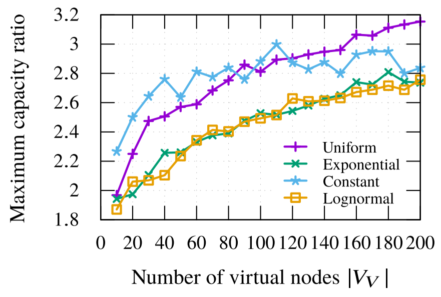

Fig. 12 Number of nodes and maximum capacity ratio of Min as found in a supplemental experiment.

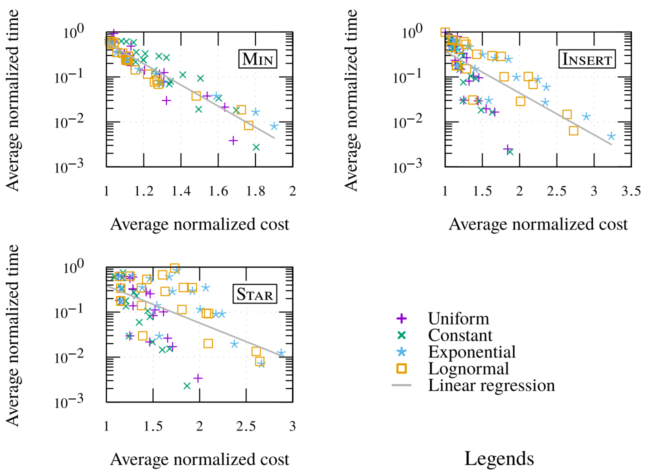

Fig. 13 Trade-off between computation time and embedding cost. Link reduction speeds up computation exponentially with a linear increase in embedding cost.

no capacity ratio above 3.2, even when the number of nodes $|V_V|$ was raised to 200. If Theorem 2 were tight then the maximum ratio should have exceeded ten thousand when $|V_V|$ is 200. However, the maximum ratio remained small even when the number of virtual nodes increased, and the bound seems very small such as log.

6.3 Trade-Off Between Time and Cost

Figure 13 shows scatter plots for the trade-off between computation time and embedding cost. Each point corresponds to a combination of three parameter values, $|V_V|$, β, and capacity distribution, as in Fig. 9. Link reduction speeds up the computation exponentially with only a linear increase in embedding cost. In Min scheme, when computation time was halved, the embedding cost increased by only 10%. Even when the time was reduced by factor of a thousand, the cost was just doubled. The results well agree with our conjecture (30). This good trade-off shows that we can reduce dense VNs for fast embedding without a significant cost increase.

To compare the trade-off between the three schemes, we analyzed with linear regression. Table 4 shows absolute values of linear regression coefficients, i.e., the slopes of the regression lines in Fig. 13. Given the same normalized cost, the larger the absolute value of regression coefficients is, the faster the computation time becomes. In this table, *All* means linear regression over all results and the remaining represent over the specific distribution. Although, when the capacity distribution is uniform or constant, the three schemes have comparable trade-off, Min scheme has the best overall trade-off among the three schemes.

Table 4 Absolute value of regression coefficients.

Scheme	All	Uniform	Constant	Exponential	Lognormal
Min	5.4	6.6	6.7	4.6	5.3
Insert	2.2	6.6	7.0	2.0	2.2
Star	1.9	5.0	6.5	2.0	2.0

7. Related Work

VNE has been well studied for a decade [2], [3]. Reference [2] broadly summarizes network virtualization including VNE. Reference [3] focuses on the VNE problem. Two types of method has been proposed to solve VNE: exact methods and approximation methods. Exact methods find the optimal embedding rely on mixed integer programming [14], [25], [26], and so struggle with large problems. On the other hand, approximation methods find *good* embedding, instead of the optimal one, in a reasonable time. Those approximation methods utilize LP relaxation [9]–[12], column generation technique [27]–[29] greedy method [6], modified subgraph isomorphism matching [7], minimum cost maximum clique [30], or meta heuristics [8], [31]. References [12], [25], [32] partition a VN into several pieces, in order to embed across multiple InPs. References [33], [34] split a PN for better embedding efficiency. However, these existing methods take a VN as given and never redesign it before the embedding process. Thus, our method complements all of them as regards reducing the embedding time.

Linear programming *relaxation* is one of the conventional techniques used to tackle with $\mathcal{NP}$-hard problems [13]. Many tools [24], [35], [36] have been developed to solve LP problems, and the LP algorithms have been well studied from the viewpoints of practicality and theory. Conventional methods for solving an LP are interior method [20], [37] and simplex method [19]. The interior method runs in $O(n^{3.5})$ where n is the number of variables in the LP. The simplex method [19] is theoretically an exponential algorithm, but it is known to run $O(n^3)$ in practice [21]. Any speed up enhancement techniques based on it complement our method since we use an LP solver as a subroutine.

Several graph operations, such as concatenation, have been defined and analyzed [38], but none of them have been studied in the context of VNE acceleration.

8. Conclusions

In this paper, we investigated a VNE preprocessing scheme to reduce a dense VN into a tractable sparse one for fast embedding. We analytically and experimentally demonstrated that the embedding time can be decreased exponentially with a linear increase in embedding cost. Since SPs are likely to request a dense VN based on a traffic matrix, our preprocessing approaches will be essential to provide VNs on demand given the current virtualized computing infrastructures.

Future works include further analysis of the capacity ratio and feasibility ratio, development of restriction types, and a theoretical investigation of the relationship among embedding cost, capacity, and feasibility.

Acknowledgments

K. Mizutani and O. Akashi performed their work while being affiliated with NTT Network Innovation Laboratories, Japan.

References

[1] N. Chowdhury and R. Boutaba, "Network virtualization: State of the art and research challenges," IEEE Commun. Mag., vol.47, no.7, pp.20–26, 2009.
[2] A. Belbekkouche, M.M. Hasan, and A. Karmouch, "Resource discovery and allocation in network virtualization," IEEE Commun. Surveys Tuts., vol.14, no.4, pp.1114–1128, 2012.
[3] A. Fischer, J. Botero, M. Till Beck, H. de Meer, and X. Hesselbach, "Virtual network embedding: A survey," IEEE Commun. Surveys Tuts., vol.15, no.4, pp.1888–1906, 2013.
[4] E. Amaldi, S. Coniglio, A.M. Koster, and M. Tieves, "On the computational complexity of the virtual network embedding problem," Electronic Notes in Discrete Mathematics, vol.52, pp.213–220, 2016.
[5] Y. Zhu and M. Ammar, "Algorithms for assigning substrate network resources to virtual network components," Proc. IEEE INFOCOM, pp.1–12, 2006.
[6] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: Substrate support for path splitting and migration," SIGCOMM Comput. Commun. Rev., vol.38, no.2, pp.17–29, 2008.
[7] J. Lischka and H. Karl, "A virtual network mapping algorithm based on subgraph isomorphism detection," Proc. ACM VISA, pp.81–88, 2009.
[8] X. Cheng, S. Su, Z. Zhang, H. Wang, F. Yang, Y. Luo, and J. Wang, "Virtual network embedding through topology-aware node ranking," SIGCOMM Comput. Commun. Rev., vol.41, no.2, pp.38–47, 2011.
[9] M. Chowdhury, M. Rahman, and R. Boutaba, "ViNEYard: Virtual network embedding algorithms with coordinated node and link mapping," IEEE/ACM Trans. Netw., vol.20, no.1, pp.206–219, 2012.
[10] C. Papagianni, A. Leivadeas, S. Papavassiliou, V. Maglaris, C. Cervello-Pastor, and A. Monje, "On the optimal allocation of virtual resources in cloud computing networks," IEEE Trans. Comput., vol.62, no.6, pp.1060–1071, June 2013.
[11] M. Rahman and R. Boutaba, "SVNE: Survivable virtual network embedding algorithms for network virtualization," IEEE Trans. Netw. Service Manage, vol.10, no.2, pp.105–118, June 2013.
[12] A. Leivadeas, C. Papagianni, and S. Papavassiliou, "Efficient resource mapping framework over networked clouds via iterated local search-based request partitioning," IEEE Trans. Parallel Distrib. Syst., vol.24, no.6, pp.1077–1086, June 2013.
[13] J. Hromkovic and W.M. Oliva, Algorithmics for Hard Problems, 2nd ed., Springer, Secaucus, NJ, USA, 2002.
[14] C. Wang and T. Wolf, "Virtual network mapping with traffic matrices," Proc. ACM/IEEE ANCS, pp.225–226, Oct. 2011.
[15] Cisco, "Visual networking index: Forecast and trends, 2017–2022," https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html
[16] E.T.S.I. (ETSI), "Network functions virtualisation (NFV); management and orchestration," Std. ETSI GS NFV-MAN 001, Dec. 2014.
[17] T. Kamada, Y. Kuno, H. Tamura, and H. Iwamiya, "Practical implementation of virtualization platform in NTT DOCOMO network," NTT DOCOMO Techinical Journal, vol.18, no.1, pp.20–28, July 2016.
[18] L. Foundation, "DPDK: Data plane development kit," http://dpdk.org/
[19] G.B. Dantzig, "Maximiztion of a linear function of variables subject to linear inequalities," Proc. Activity Analysis of Production and Allocation, New York, pp.339–347, 1951.
[20] N. Karmarkar, "A new polynomial-time algorithm for linear programming," Combinatorica, vol.4, no.4, pp.373–396, 1984.
[21] M.J. Todd, "The many facets of linear programming," Math. Program., vol.91, no.3, pp.417–436, 2002.
[22] O. Goldschmidt and D.S. Hochbaum, "A polynomial algorithm for the k-cut problem for fixed k," Math. Oper. Res., vol.19, no.1, pp.24–37, Feb. 1994.
[23] H. Saran and V.V. Vazirani, "Finding k-cuts within twice the optimal," SIAM J. Comput., vol.24, no.1, pp.101–108, Feb. 1995.
[24] GNU, "GNU Linear Programming Kit," http://www.gnu.org/software/glpk/
[25] I. Houidi, W. Louati, W.B. Ameur, and D. Zeghlache, "Virtual network provisioning across multiple substrate networks," Computer Networks, vol.55, no.4, pp.1011–1023, 2011.
[26] S.R. Chowdhury, R. Ahmed, M.M.A. Khan, N. Shahriar, R. Boutaba, J. Mitra, and F. Zeng, "Dedicated protection for survivable virtual network embedding," IEEE Trans. Netw. Serv. Manag., vol.13, no.4, pp.913–926, 2016.
[27] Q. Hu, Y. Wang, and X. Cao, "Resolve the virtual network embedding problem: A column generation approach," 2013 Proc. IEEE INFOCOM, p.nil, April 2013.
[28] A. Jarray and A. Karmouch, "Decomposition approaches for virtual network embedding with one-shot node and link mapping," IEEE/ACM Trans. Netw., vol.23, no.3, pp.1012–1025, 2015.
[29] R. Mijumbi, J. Serrat, J.L. Gorricho, and R. Boutaba, "A path generation approach to embedding of virtual networks," IEEE Trans. Netw. Serv. Manag., vol.12, no.3, pp.334–348, Sept. 2015.
[30] L. Gong, H. Jiang, Y. Wang, and Z. Zhu, "Novel location-constrained virtual network embedding lc-vne algorithms towards integrated node and link mapping," IEEE/ACM Trans. Netw., vol.24, no.6, pp.3648–3661, 2016.
[31] S. Su, Z. Zhang, A.X. Liu, X. Cheng, Y. Wang, and X. Zhao, "Energy-aware virtual network embedding," IEEE/ACM Trans. Netw., vol.22, no.5, pp.1607–1620, 2014.
[32] T.H. Lee, S. Tursunova, and T.S. Choi, "Graph clustering based provisioning algorithm for virtual network embedding," Proc. IEEE NOMS, pp.1175–1178, April 2012.
[33] T. Ghazar and N. Samaan, "Hierarchical approach for efficient virtual network embedding based on exact subgraph matching," Proc. IEEE GLOBECOM, pp.1–6, Dec. 2011.
[34] F. Yang, Z. kai Wang, J. ya Chen, and Y. jie Liu, "VLB-VNE: A regionalized valiant load-balancing algorithm in virtual network mapping," Proc. IEEE WCNIS, pp.432–436, June 2010.
[35] IBM, "CPLEX Optimizer," https://www.ibm.com/analytics/cplex-optimizer
[36] GUROBI, "Gurobi Optimizer," http://www.gurobi.com/
[37] Y. Ye, Interior Point Algorithms: Theory and Analysis, John Wiley & Sons, New York, NY, USA, 1997.
[38] D.B. West, Introduction to Graph Theory, Prentice Hall, Upper Saddle River, 2000.

Appendix: Proofs

A.1 Proof of Corollary 9

First, we prove the upper bound $2^{O(|V|)}$ by using Theo-

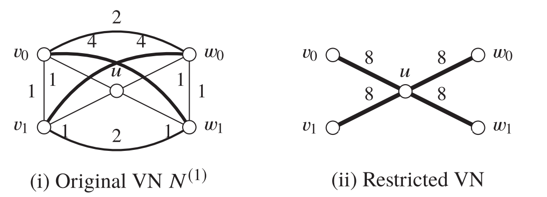

Fig. A· 1 An example of VN whose feasibility ratio is at least 2.

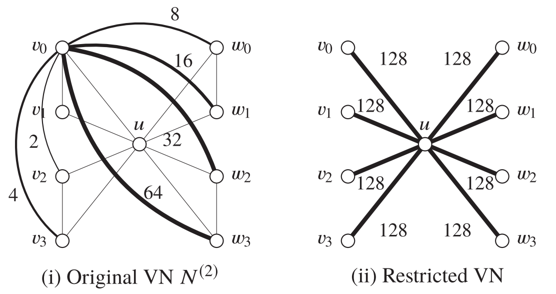

Fig. A· 2 An example of VN whose feasibility ratio is at least 16.

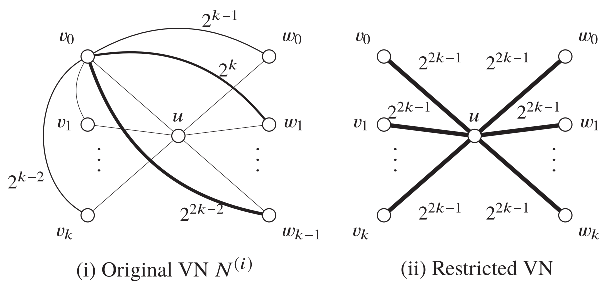

Fig. A· 3 An example of VN example whose feasibility ratio is $2^{\Omega(|V|)}$ where $k = 2^i$.

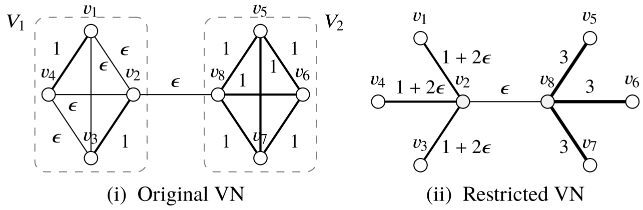

Fig. A· 4 An example of VN whose feasibility ratio is at least $1/2\epsilon$. We omitted links that has zero required capacity and filled core nodes in gray.

rem 7. Then, leveraging Theorem 8, we prove the lower bound $2^{\Omega(|V|)}$ by constructing a VN that has the desired feasibility ratio; We begin with a small example that has a relatively large feasibility ratio. Then, we construct a VN by using the small example as building blocks and show that has the desired feasibility ratio.

We begin with the upper bound. For any link $e \in E$, the number of link reductions in which the reduced triangle contains link e is at most $|V| - 2$. This is because the number of triangles that contain link e is $|V| - 2$. Thus, for any link $e \in E'$ in restricted VN N', the required capacity after multiple minimum capacity link reductions, $r'(e)$, is bounded above by $2^{|V|-2} r(e)$. Thus, the feasibility ratio is $2^{O(|V|)}$ from Theorem 7.

We move on to the lower bound. Let $N^{(1)}$ be a VN that has five nodes $V = \{u, v_0, v_1, w_0, w_1\}$, the link capacity of $\{v_0, w_1\}$ and $\{v_1, w_0\}$ be 4, that of $\{v_0, w_0\}$ and $\{v_1, w_1\}$ be 2, and the remaining link capacity be 1 (Fig. A· 1(i)). Reducing minimum capacity links in the following order, we obtain the restricted network that has star topology with center u and all link capacities are 8 (Fig. A· 1(ii)): First, we reduce links $\{v_0, v_1\}$ and $\{w_0, w_1\}$ to triangles (u, v_0, v_1) and (u, w_1, w_2). Then, we reduce links $\{v_0, w_0\}$ and $\{v_1, w_1\}$ to triangles that contain node u as previous links. Next, we reduce links $\{v_0, w_1\}$, $\{v_1, w_0\}$ in the same manner. Hence, Theorem 8 with $k = 1$ yields that the feasibility ratio is at least $2 = 8/4$.

Let $N^{(2)}$ be a VN that is made up of two copies of $N^{(1)}$ and the links between them (Fig. A· 2(i)). One of the copies is $\{u, v_0, v_1, v_2, v_3\}$ and the other is $\{u, w_0, w_1, w_2, w_3\}$. Links between v_i and $w_{(i+j) \bmod 4}$ have 2^{3+j} required capacity for $i = 0, 1, 2, 3$ and $j = 0, 1, 2, 3$. Note that, Fig. A· 2(i) shows only links that has 1 capacity or that connect to node v_0. We omitted the other links for the sake of visibility. As in Fig. A· 1, reducing minimum capacity links to triangles that contain center node u yields the restricted network that has star topology with center u and all link capacities are 128 (Fig. A· 2(ii)). The feasibility ratio is at least $16 = 128/8$ in the same manner as $N^{(1)}$.

By recursively repeating the above construction $i - 1$ times, we gain the VNs of Fig. A· 3(i) where $k = 2^i$. Note that, Fig. A· 3(i) shows links that connect to either node u or v_0 and omits the other links as well as the labels of links whose required capacity is 1. After reducing minimum capacity links to triangles that contain center node u, we obtain the restricted one of Fig. A· 3(ii). Again, Theorem 8 with $k = 1$ yields that the feasibility ratio is at least

$$\frac{2^{2k-1}}{2k} = 2^{2^{i+1}-i-2} = 2^{|V|-\log(|V|-1)-2}. \quad \text{(A· 1)}$$

Thus, the feasibility ratio is $2^{\Omega(|V|)}$.

A.2 Proof of Corollary 11

Let N be a VN that has eight nodes $V = \{v_1, v_2, \ldots, v_8\}$ and each link $e = \{v_i, v_j\}$ has capacity r as in Fig. A· 4(i). Let $\{V_1, V_2\}$ be a node partition such that $V_1 = \{v_1, \ldots, v_4\}$ and $V_2 = \{v_5, \ldots, v_8\}$. Let N' be the restricted network by Insert with respect to the node partition $\{V_1, V_2\}$ and core nodes $\{v_2, v_8\}$ (Fig. A· 4(ii)). Note that, the node partition $\{V_1, V_2\}$ is a minimum cut and core nodes, v_2 and v_8, have maximum connected link capacity; connected link capacity of v_2 is $1 + 3\epsilon$ and that of v_8 is $3 + \epsilon$.

We apply Theorem 8 with $k = 2$ to these VNs. The two nodes $\{v_1, v_4\}$ minimize the connected link capacity in the original VN and that of the restricted VN. Hence, the feasibility ratio is at least $(2 + 4\epsilon)/4\epsilon = 1 + 1/2\epsilon$.

A.3 Proof of Corollary 13

Let N be the left side of VN in the proof of Corollary 11 (Fig. A· 4(i)), that is, N has four virtual nodes

$V = \{v_1, v_2, v_3, v_4\}$ and each link e has the following link capacity: if link e is either $\{v_1, v_4\}$ or $\{v_2, v_3\}$ then the capacity $r(e)$ is 1, otherwise ϵ. Let N' be the restricted VN yielded by Star with $s = 3$. Let's say node v_1 be chosen as the center node. Then, VN N' has three virtual links $\{v_1, v_2\}$, $\{v_2, v_3\}$, and $\{v_1, v_4\}$ and these links have the same capacity $1 + 2\epsilon$.

We apply Theorem 8 with $k = 2$ to these VNs. In the same manner as used to prove Corollary 11, the minimum connected link capacity in the original VN N is 4ϵ and that of the restricted VN N' is $2 + 4\epsilon$. Hence, the feasibility ratio is at least $1 + 1/2\epsilon = (2 + 4\epsilon)/4\epsilon$.

Toru Mano is a researcher in NTT Network Innovation Labs. He received the B.E. and M.E. degrees from the University of Tokyo in 2009 and 2011, respectively. His research interests are network architectures and network optimization. He is a member of ORSJ.

Takeru Inoue received the B.E. and M.E. degrees in engineering science and the Ph.D. degree in information science from Kyoto University, Kyoto, Japan, in 1998, 2000, and 2006, respectively. He is a Senior Researcher with Nippon Telegraph and Telephone Corporation Laboratories, Tokyo, Japan. He was an ERATO Researcher with the Japan Science and Technology Agency, Kawaguchi, Japan, from 2011 to 2013. His research interests widely cover algorithmic approaches in computer networks. He was a recipient of the Best Paper Award of the Asia-Pacific Conference on Communications in 2005. He is a member of IEEE.

Kimihiro Mizutani received the M.S. and Ph.D. degrees from the Nara Institute of Science and Technology, in 2010 and 2015, respectively. He was a researcher at NTT Group (Network Innovation Labs and West R&D Center) from 2010 to 2019. Currently, he is a lecture (Principal Investigator) in the department of informatics at Kindai university. His research interests include future network architectures and various systems powered by deep learning. He was a recipient of the Best Student Paper Award from ICCSA and the Research Awards from IEICE in 2010 and 2013, respectively. He is a member of IEEJ.

Osamu Akashi received the B.Sc. and M.Sc. degrees in information science and the Ph.D. degree in mathematical and computing sciences from the Tokyo Institute of Technology, in 1987, 1989, and 2001, respectively. He was a Senior Research Scientist in Network Innovation Laboratories with Nippon Telegraph and Telephone Corporation to 2018. Currently, he is a Research Professor at National Institute Informatics, Japan. His research interests are in distributed systems, network management, and network architecture. He is a member of ACM, IPSJ, and JSSST.