Computer Networks 112 (2017) 52–66

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Virtual network embedding with multiple priority classes sharing substrate resources

Nagao Ogino[a,*], Takeshi Kitahara[a], Shin'ichi Arakawa[b], Masayuki Murata[b]

[a] Communications Network Planning Laboratory, KDDI R&D Laboratories Inc. 2-1-15 Ohara, Fujimino, Saitama 356-8502 Japan
[b] Graduate School of Information Science and Technology, Osaka University, 1-5 Yamadaoka, Suita, Osaka 565-0871 Japan

ARTICLE INFO

Article history:
Received 15 June 2016
Revised 4 October 2016
Accepted 6 October 2016
Available online 27 October 2016

Keywords:
Network virtualization
Virtual network embedding
Multiple priority classes
Substrate resource sharing
Effect of statistical multiplexing

ABSTRACT

Network virtualization is a promising approach for virtual network operators to configure their own inter-cloud networks flexibly on an inter-cloud substrate network. Virtual network operators specify multiple priority classes in order to satisfy different latency requirements for their inter-cloud network services with a variety of delay-sensitivities on their virtual networks. Furthermore, they may require substrate resource sharing among multiple priority classes in order to reduce the amount of substrate resources assigned to them. Meanwhile, it is desirable for a substrate network provider that multiple virtual networks can share substrate resources since the total amount of substrate resources required can be reduced due to the effect of statistical multiplexing. This paper formulates a novel virtual network embedding problem and proposes a heuristic virtual network embedding method to minimize the total substrate resources required when the virtual network operators request substrate resource sharing among multiple priority classes within their virtual networks. Based on the proposed method, multiple virtual networks can maximally share substrate resources with one another while sharing substrate resources on an equal basis. The effect of substrate resource sharing among multiple priority classes and multiple virtual networks is quantitatively assessed through extensive simulations, and advantageous topologies for requested virtual networks and substrate networks are also presented.

1. Introduction

Network virtualization is a promising approach to mitigate the rigidity of the current Internet [1–3]. It allows virtual network operators to configure their own inter-cloud networks flexibly on a substrate network connecting multiple cloud systems. Each virtual network operator can provide its inter-cloud network service using the virtual network composed of virtually isolated substrate resources. The virtual network operators may prepare multiple types of inter-cloud network services with different delay-sensitivities simultaneously on their virtual networks. Aiming to satisfy different latency requirements for the inter-cloud network services, the virtual network operators specify multiple priority classes for data transfer via virtual links and service processing on virtual nodes within their virtual networks. Furthermore, the virtual network operators may require substrate resource sharing among the multiple priority classes in order to utilize efficiently the substrate resources assigned to them.

One of the significant problems for a substrate network provider is how to embed the requested virtual networks into the substrate network, and a variety of virtual network embedding (VNE) methods have been proposed [4–20]. However, these methods do not take into consideration substrate resource sharing among the multiple priority classes within each of the requested virtual networks. Furthermore, if multiple virtual network operators that require substrate resource sharing among their different priority classes can share substrate resources fairly, the total required amount of substrate resources can be reduced due to the effect of statistical multiplexing among multiple virtual networks. Addressing the above, this paper mathematically formulates a novel VNE problem and proposes a heuristic VNE method in order to minimize the total amount of substrate resources required when each virtual network operator needs to share substrate resources among multiple priority classes.

First, this paper presents a model to quantify the amount of substrate resources required while considering different latency requirements for multiple priority classes and the effect of statistical multiplexing. This paper also examines the condition where

* Corresponding author.
E-mail addresses: ogino@kddilabs.jp (N. Ogino), kitahara@kddilabs.jp (T. Kitahara), arakawa@ist.osaka-u.ac.jp (S. Arakawa), murata@ist.osaka-u.ac.jp (M. Murata).

http://dx.doi.org/10.1016/j.comnet.2016.10.007
1389-1286/

the substrate resources can be shared fairly among multiple virtual networks. Based on the above consideration, this paper formulates a VNE problem using an integer programming model. To solve the formulated VNE problem efficiently, this paper proposes a heuristic VNE method to minimize the total amount of substrate resources required. The proposed method can maximize fair substrate resource sharing among multiple virtual network operators. Simulation results demonstrate that the proposed heuristic method can efficiently achieve sub-optimum VNE satisfying the above condition.

Section 2 explains related work as the background of this paper. Section 3 illustrates the considered VNE problem. Section 4 proposes a heuristic VNE method to solve the above problem. Section 5 presents several results of simulation experiments. Finally, Section 6 concludes this paper.

2. Related work on virtual network embedding

Generally, the virtual network embedding (VNE) problem can be formulated using the integer programming model. Since the VNE problem is *NP*-hard, a variety of heuristic methods have been proposed [4]. Most of these heuristic methods solve the virtual node and virtual link assignment separately. First, each virtual node is assigned to a substrate node using the heuristics related to the resource utilization and topology information around each substrate node [5–10]. Although each virtual link is generally assigned to the shortest substrate path between a pair of substrate nodes to which the virtual nodes are assigned, a virtual link may be assigned to multiple substrate paths by solving the multi-commodity flow problem [11,12]. In contrast, several heuristic methods assign the virtual node and virtual link simultaneously. Coordination between the virtual node and virtual link assignment was attempted by relaxing the integer programming model [12] and using the heuristics derived from the overall topology of the virtual and substrate networks [13,14]. Each virtual node is progressively assigned together with its incident virtual links according to the residual substrate node capacity and residual substrate link bandwidth [15]. Each virtual node may be assigned to multiple substrate nodes [16]. A formula using the integer programming model was also devised to solve the VNE problem efficiently [17]. Recently, several heuristic VNE methods over elastic optical networks have been proposed [18]. A reliable virtual network design method and VNE method to survive a facility node failure have also been proposed [19]. An energy efficient VNE method for cloud computing was formulated using a mixed integer programming model [20].

Zhang et al. [7,8] have considered substrate resource sharing among multiple virtual networks. However, their method only considers substrate resource sharing within the same priority class and does not consider sharing among different priority classes while at the same time satisfying the different latency requirements. Thus, the method cannot be applied to the VNE problem where substrate resource sharing among multiple priority classes is required within each requested virtual network. This paper proposes a heuristic VNE method to minimize the required amount of substrate resources due to fair substrate resource sharing among multiple virtual networks, while considering the existence of multiple priority classes that share the substrate resources with one another within each virtual network. Since substrate resource sharing among multiple virtual networks is expected to occur on the substrate link bandwidth more frequently, the proposed heuristic method prioritizes virtual link assignment rather than virtual node assignment, in contrast to most of the existing heuristic methods.

3. Virtual network embedding problem

This section explains the virtual network embedding (VNE) problem discussed in this paper. First, the considered model for the requested virtual networks is presented. A model to quantify the required amount of substrate resources and the effect of statistical multiplexing caused by substrate resource sharing is shown, and the condition whereby substrate resources can be shared fairly among multiple virtual networks is clarified. Finally, the considered VNE problem is formulated using an integer programming model.

3.1. Virtual network model

The inter-cloud substrate network is composed of multiple substrate nodes, i.e., cloud systems, and substrate links connecting a pair of substrate nodes, i.e., inter-cloud links. Here, each cloud system represents a datacenter composed of multiple servers or a network function virtualization infrastructure point of presence (NFVI-PoP) [21]. This means that each substrate node also involves switching nodes and each substrate link connects a pair of switching nodes involved in a pair of substrate nodes. Multiple virtual network operators request VNE in order to provide their inter-cloud network services on their virtual networks. Each virtual network operator also specifies an appropriate priority class for each type of the inter-cloud network service with different delay sensitivities. Thus, each virtual network request includes the following specification.

1) Topology of the virtual network;
2) Priority class that each virtual link belongs to;

The topology of the virtual network indicates a set of virtual links that connect pairs of virtual nodes. The priority class to which each virtual link belongs indicates the priority class for data transfer through the virtual link. A pair of service components is assigned to two virtual nodes terminating each virtual link. The pair of service components belongs to the same priority class as the corresponding virtual link and is executed on the two virtual nodes according to the specified priority class. This means that each virtual link and pair of service components at both ends of the virtual link comprise one module of inter-cloud network services, for which a priority class is specified.

When the requested virtual networks are embedded into the substrate network, the substrate nodes to which each virtual node can be assigned may be restricted due to the latency constraint on the end terminals to access the nearby virtual nodes [12]. This paper assumes that each virtual node is assigned to the specified central substrate node or one of the substrate nodes adjacent to the central substrate node. Two virtual nodes in a virtual network cannot be assigned to an identical substrate node. Each virtual link is assigned to one of the substrate paths connecting two substrate nodes to which a pair of virtual nodes terminating the virtual link is assigned. This means that each virtual link may traverse multiple substrate links and multiple virtual links may traverse an identical substrate link. Multiple virtual links traversing an identical substrate link are requested to share the substrate link bandwidth with each other. In the same way, multiple service components executed on a virtual node are requested to share the substrate node capacity assigned to the virtual node.

Fig. 1 shows an example of VNE considered in this paper. Fig. 1(a) shows the virtual network to be embedded. The virtual network includes four virtual nodes and four virtual links. Two virtual links *vl1* and *vl4* belong to priority class *pr1* and the other two virtual links *vl2* and *vl3* belong to priority class *pr2*. Fig. 1(a) also shows service components executed on each virtual node. For example, virtual node *vn6* executes one service component belonging

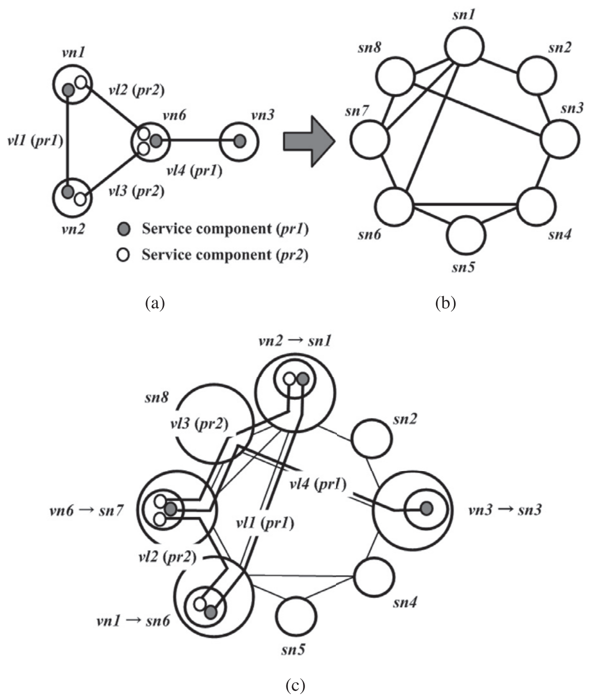


Fig. 1. Considered virtual network embedding model. (a) Example of requested virtual network. (b) Example of substrate network. (c) Example of virtual network embedding.

to priority class *pr1* and two service components belonging to priority class *pr2*. Fig. 1(b) shows a substrate network into which the virtual network is embedded. The substrate network includes eight substrate nodes and twelve substrate links. In Fig. 1, a common number is attached to a virtual node and its central substrate node. The central substrate nodes for virtual nodes *vn1, vn2, vn3*, and *vn6* are *sn1, sn2, sn3*, and *sn6*, respectively. Fig. 1(c) shows an example of VNE on the substrate network. In Fig. 1(c), virtual nodes *vn1, vn2, vn3*, and *vn6* are assigned to *sn6, sn1, sn3*, and *sn7*, respectively. Each virtual node is assigned to its central substrate node or one of the substrate nodes adjacent to the central substrate node. In this VNE, virtual link *vl3* traverses two substrate links *sn7-sn8* and *sn8-sn1*, and virtual link *vl4* traverses two substrate links *sn7-sn8* and *sn8-sn3*. This means that two virtual links *vl3* and *vl4* belonging to the different priority classes traverse a common substrate link *sn7-sn8* and are requested to share the bandwidth of substrate link *sn7-sn8* with each other.

3.2. Model to assess required amount of substrate resource

In this subsection, the required amount of substrate resources is related to the specified priority classes and latency requirements for the inter-cloud network services that the virtual network operators provide. The required amount of substrate resources is calculated using the following model while considering the effect of statistical multiplexing. First, the traffic volume in a virtual link and a service component is denoted by the notation T. The traffic volume in a virtual link is equivalent to the data volume that the virtual link conveys through the substrate links during a unit of time, and the traffic volume in a service component is equivalent to the work volume that the service component imposes on the substrate node during a unit of time. The amount of substrate resource B assigned for a virtual link and a service component can be expressed as follows [22,23]:

$$\mathrm{B} = \overline{T} + k\sigma(T) \tag{1}$$

Table 1
Queuing delay for each traffic volume.

Average traffic volume	Value of k	Amount of substrate resource	Queuing delay
$\overline{T1}=0.50$	$k_1=1.061$	$B_1=1.250$	$W_1=0.267$
$\overline{T2}=0.50$	$k_2=0.354$	$B_2=0.750$	$W_2=1.333$
$\overline{T}=1.00$	$k=0.707$	$B=1.707$	$W_1=0.243$
			$W_2=0.586$

In the above expression, the notations $\overline{T}$ and $\sigma(\mathrm{T})$ indicate the average and standard deviation of the traffic volume T, respectively. The first term in the above expression indicates the average part of the amount of substrate resources, which cannot be shared with any other virtual links or service components. In contrast, the second term indicates the deviation part, which can be shared with other virtual links or service components if necessary.

The queuing delay of packets transferred through a virtual link can be reduced as the substrate link bandwidth assigned for the virtual link increases. Likewise, the queuing delay of transactions to a service component decreases as the substrate node capacity assigned for the service component increases. This means that the increase in the amount of substrate resources assigned can achieve smaller latency. Thus, a higher priority class is specified for inter-cloud network services with higher delay-sensitivity and a larger value of k is specified for the virtual links and service components belonging to the higher priority class. Table 1 shows an example of the queuing delay in two virtual links or service components belonging to different priority classes, which are calculated using an M/D/1 queuing model [24]. The traffic volumes in two virtual links and service components are denoted by the notations T_i ($i=1, 2$), respectively. Although the average traffic volumes $\overline{Ti}$ ($i=1, 2$) are identical, different values of k_i ($i=1, 2$) are specified for the two virtual links or service components. Here, the variances in the traffic volumes $\sigma^2(\mathrm{Ti})$ ($i=1, 2$) are identical to the average traffic volumes $\overline{Ti}$ since the Poisson arrival process of the packets and transactions is assumed in Table 1. As shown in Table 1, the greater value of k reduces the queuing delay due to an increase in the assigned amount of substrate resources.

The amount of substrate resources shared among multiple virtual links and service components can be given as follows. When the aggregated traffic volume in multiple virtual links and service components is denoted by the notation T (= $\Sigma\mathrm{Ti}$), the average of the aggregated traffic volume $\overline{T}$ is merely given by summing the average of the individual traffic volumes $\overline{Ti}$. The variance of the aggregated traffic volume $\sigma^2(\mathrm{T})$ is given by summing the variance of the individual traffic volumes $\sigma^2(\mathrm{Ti})$, when the traffic volume in each virtual link and service component is independent of the others. Thus, the assigned amount of substrate resources for multiple virtual links and multiple service components (B) can be approximated as follows when these virtual links and service components share the substrate resources:

$$B = \overline{T} + k\sigma(T) = \overline{\sum_i Ti} + k\sigma\Big(\sum_i Ti\Big)$$
$$\cong \sum_i \overline{Ti} + k\sqrt{\sum_i \sigma^2(Ti)} \tag{2}$$

For example, the value of k is given by the average value of k_i weighted by the average traffic volume $\overline{Ti}$ as follows:

$$k = \sum_i ki\overline{Ti} / \sum_i \overline{Ti} \tag{3}$$

In contrast, the amount of substrate resources assigned to multiple virtual links and service components (B') simply becomes the total sum of the amount of substrate resources required for each

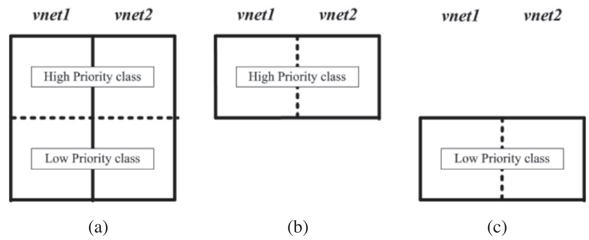


Fig. 2. Example of substrate link bandwidth sharing among multiple virtual networks. (a) Multiple virtual links having high and low priorities traverse in *vnet1* and *vnet2*. (b) Only high priority virtual links traverse in *vnet1* and *vnet2*. (c) Only low priority virtual links traverse in *vnet1* and *vnet2*.

virtual link and service component when substrate resource sharing is not permitted:

$$B' = \sum_i \overline{Ti} + \sum_i ki\sigma\,(Ti) \qquad (4)$$

Table 1 also shows the queuing delay when two virtual links that convey the traffic volumes T_i ($i = 1, 2$) share the substrate link bandwidth or two service components that impose the traffic volumes T_i ($i = 1, 2$) share the substrate node capacity. The amount of substrate resource B assigned to the two virtual links and two service components is given by expressions (2) and (3). The queuing delay for each of the traffic volumes W_i ($i = 1, 2$) is calculated using the M/D/1 HOL queuing model [24]. This means that the two types of traffic volumes in different priority classes utilize the shared substrate resources according to the priority queuing discipline [25]. As shown in Table 1, the queuing delay for each traffic volume can be maintained due to the effect of statistical multiplexing although the total amount of substrate resources shared between the two virtual links and service components decreases.

3.3. Policy of fair substrate resource sharing

Aiming to reduce the assigned amount of substrate resources, each virtual network operator prioritizes sharing of the substrate resources among its multiple virtual links and service components. Multiple virtual links traversing the same substrate link are required to share the substrate link bandwidth even if those multiple virtual links belong to different priority classes. Likewise, multiple service components on the same virtual node are requested to share the substrate node capacity even when the multiple service components belong to different priority classes. The utilization of the substrate resources that multiple virtual links or service components share amongst each other can be scheduled on the basis of the strict priority queuing mechanism [25].

If multiple virtual network operators can share substrate resources, more efficient utilization of substrate resource can be achieved. Fig. 2 schematically explains the policy of substrate link bandwidth sharing between two virtual networks *vnet1* and *vnet2*. Fig. 2 depicts a boundary between two regions as a solid line if the substrate link bandwidth expressed by the two regions should be isolated. In contrast, the boundary is a broken line if the substrate link bandwidth expressed by the two regions can be shared between the two priority classes or the two virtual networks. In Fig. 2(a), *vnet1* and *vnet2* include multiple virtual links of high and low priorities and the multiple virtual links share the substrate link bandwidth within each of *vnet1* and *vnet2*. If *vnet1* and *vnet2* share the substrate link bandwidth between each other, the low priority virtual link in *vnet1* could be throttled to ensure small latency for the high priority virtual link in *vnet2* and vice versa, in the event of congestion. From the perspective of fairness, each virtual network operator prohibits its low priority virtual links from being sacrificed for high priority virtual links in the other virtual networks. This means that a virtual network, where virtual links included in multiple priority classes traverse the considered substrate link, cannot share substrate link bandwidth with the other virtual networks.

Fig. 2(b) shows a case where only high priority virtual links traverse the considered substrate link in both *vnet1* and *vnet2*. Even if *vnet1* and *vnet2* share the substrate link bandwidth, the unilateral of sacrifice of virtual links in either *vnet1* or *vnet2* is prevented in the event of congestion. Meanwhile, the substrate link bandwidth assigned for both *vnet1* and *vnet2* can be reduced due to the effect of statistical multiplexing. Fig. 2(c) shows a similar case where only low priority virtual links traverse the considered substrate link in *vnet1* and *vnet2*. Even if *vnet1* and *vnet2* share the substrate link bandwidth, virtual links in neither *vnet1* nor *vnet2* are throttled unilaterally in the event of congestion. Instead, the substrate link bandwidth assigned for *vnet1* and *vnet2* is utilized efficiently due to the effect of statistical multiplexing. Each virtual network operator permits the substrate link bandwidth to be shared with the other operators on an equal basis. This means that multiple virtual networks, where only virtual links of the same priority traverse the considered substrate link, can share the substrate link bandwidth.

The same argument also holds for the substrate node capacity. A virtual node that executes service components belonging to multiple priority classes cannot share the substrate node capacity with the other virtual nodes. In contrast, a virtual node that only executes service components in one priority class can share the substrate node capacity with other virtual nodes that only executes service components in the same priority class. Since most of the virtual nodes execute service components belonging to multiple priority classes, substrate resource sharing among multiple virtual networks is likely to occur on the substrate links rather than on the substrate nodes.

3.4. Formulation of virtual network embedding problem

The considered VNE problem minimizes the total amount of substrate resources required after each of the requested virtual networks has been embedded. This means that the reassignment of the existing virtual networks is not taken into account. The average amount of substrate resources assigned to each virtual network is also minimized by solving the considered VNE problem. The considered VNE problem can be formulated using an integer programming model as follows [4,7,12,17,19,20].

First, the substrate network is extended according to the constraint on the assignment of the virtual nodes to the substrate nodes. An extended node that represents each virtual node is added to the substrate network. Each extended node is connected to its central substrate node and all the substrate nodes adjacent to the central substrate node using extended links. In the extended substrate network, each virtual link is established between a pair of extended nodes that represent two virtual nodes terminating the virtual link. The extended link traversed by virtual links indicates the mapping of each virtual node to the substrate node. Two extended links starting from an identical extended node cannot be traversed by virtual links since each virtual node is only assigned to a substrate node. Similarly, two extended links terminating at a substrate node cannot be traversed by virtual links since two virtual nodes cannot be assigned to an identical substrate node. Fig. 3 illustrates the extended substrate network for the substrate network shown in Fig. 1(b), when the virtual network in Fig. 1(a) is requested. The VNE solution shown in Fig. 1(c) allocates virtual links to four extended links *en1-sn6, en2-sn1, en3-sn3*, and *en6-sn7*.

Table 2 defines the notations for constants and sets used in the integer programming model, and Table 3 defines the notations for the variables. The constraints in the model are given as follows. First, the following constraints hold as the route preservation rule

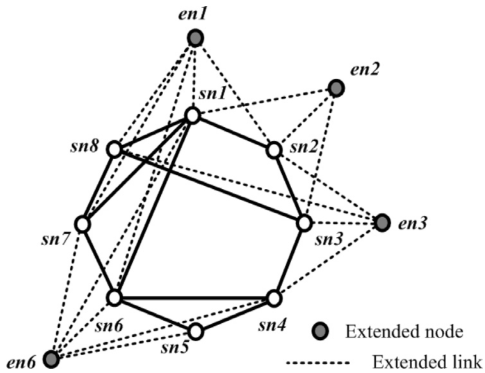


Fig. 3. Example of extended substrate network.

Table 2
Notations for constants and sets.

Notation	Definition
l	Link.
SL	Set of substrate links.
EL	Set of extended links.
n	Node.
SN	Set of substrate nodes.
EN	Set of extended nodes.
Inout (*n*)	Set of incoming and outgoing links to and from node *n*.
vnet	Existing virtual network.
VNet	Set of existing virtual networks.
pr	Priority class.
Pr	Set of priority classes.
$EQ_{pr,\ vnet}$ (*l*)	Binary number indicating whether at least one virtual link belonging to the priority class *pr* traverses the substrate link *l* in the existing virtual network *vnet* (=1) or not (=0).
$EQ1_{vnet}$ (*l*)	Binary number indicating whether at least one virtual link traverses the substrate link *l* in the existing virtual network *vnet* (=1) or not (=0).
$EQ2_{vnet}$ (*l*)	Binary number indicating whether multiple virtual links belonging to the different priority classes traverse the substrate link *l* in the existing virtual network *vnet* (=1) or not (=0).
$ENQ_{pr,\ vnet}$ (*l*)	Binary number indicating whether only virtual links belonging to the priority class *pr* traverses the substrate link *l* in the existing virtual network *vnet* (=1) or not (=0).
$EQ_{pr,\ vnet}$ (*n*)	Binary number indicating whether at least one service component belonging to the priority class *pr* is executed in the substrate node *n* in the existing virtual network *vnet* (=1) or not (=0).
$EQ1_{vnet}$ (*n*)	Binary number indicating whether at least one service component is executed in the substrate node *n* in the existing virtual network *vnet* (=1) or not (=0).
$EQ2_{vnet}$ (*n*)	Binary number indicating whether multiple service components belonging to the different priority classes are executed in the substrate node *n* in the existing virtual network *vnet* (=1) or not (=0).
$ENQ_{pr,\ vnet}$ (*n*)	Binary number indicating whether only service components belonging to the priority class *pr* are executed in the substrate node *n* in the existing virtual network *vnet* (=1) or not (=0).
Vl	Virtual link in the requested virtual network.
VL	Set of virtual links in the requested virtual network.
VL (*pr*)	Set of virtual links belonging to the priority class *pr* in the requested virtual network.
s (*vl*)	Extended node for the virtual node from which the virtual link *vl* starts.
d (*vl*)	Extended node for the virtual node to which the virtual link *vl* terminates.
A	Constant with a sufficiently large value.

for each virtual link:

$$\forall vl \in VL \quad \sum_{l \in Inout(s(vl))} Xvl, l = 1; \quad \sum_{l \in Inout(d(vl))} Xvl, l = 1; \quad \sum_{l \in Inout(n)} Xvl, l = 2 \times Yvl, n; \qquad \forall n \in SN \tag{5}$$

Table 3
Notations for variables.

Notation	Definition
$X_{vl,\ l}$	Binary variable indicating whether a virtual link *vl* in the requested virtual network traverses the substrate or extended link *l* (=1) or not (=0).
$Y_{vl,\ n}$	Binary variable indicating whether a virtual link *vl* in the requested virtual network passes through the substrate node *n* (=1) or not (=0).
Z_l	Binary variable indicating whether a virtual link in the requested virtual network traverses the extended link *l* (=1) or not (=0).
Q_{pr} (*l*)	Binary variable indicating whether at least one virtual link belonging to the priority class *pr* traverses the substrate link *l* in the requested virtual network (=1) or not (=0).
Q1 (*l*)	Binary variable indicating whether at least one virtual link traverses the substrate link *l* in the requested virtual network (=1) or not (=0).
Q2 (*l*)	Binary variable indicating whether multiple virtual links belonging to the different priority classes traverse the substrate link *l* in the requested virtual network (=1) or not (=0).
NQ_{pr} (*l*)	Binary variable indicating whether only virtual links belonging to the priority class *pr* traverses the substrate link *l* in the requested virtual network (=1) or not (=0).
Q_{pr} (*n*)	Binary variable indicating whether at least one service component belonging to the priority class *pr* is executed in the substrate node *n* in the requested virtual network (=1) or not (=0).
Q1 (*n*)	Binary variable indicating whether at least one service component is executed in the substrate node *n* in the requested virtual network (=1) or not (=0).
Q2 (*n*)	Binary variable indicating whether multiple service components belonging to the different priority classes are executed in the substrate node *n* in the requested virtual network (=1) or not (=0).
NQ_{pr} (*n*)	Binary variable indicating whether only service components belonging to the priority class *pr* are executed in the substrate node *n* in the requested virtual network (=1) or not (=0).
NNQ_{pr} (*l*)	Binary variable indicating whether only virtual links belonging to the priority class *pr* traverses the substrate link *l* in one of the existing and requested virtual networks (=1) or not (=0).
NNQ_{pr} (*n*)	Binary variable indicating whether only service components belonging to the priority class *pr* are executed in the substrate node *n* in one of the existing and requested virtual networks (=1) or not (=0).

The following constraint holds from the definition of the variable Z_l:

$$-A \times (1 - Zl) + 1 \le \sum_{vl \in VL} Xvl, l \le A \times Zl; \qquad \forall l \in EL \tag{6}$$

Since each virtual node is assigned to one substrate node and only one virtual node is assigned to each substrate node, the following constraint holds:

$$\sum_{l \in Inout(n) \cap EL} Zl \le 1; \forall n \in SN \cup EN \tag{7}$$

From the definition of the variables $Q_{pr}(l)$, $Q1(l)$, $Q2(l)$, and $NQ_{pr}(l)$, the following constraints hold for each substrate link:

$$-A \times (1 - Qpr(l)) + 1 \le \sum_{vl \in VL(pr)} Xvl, l \le A \times Qpr(l); \quad \forall pr \in Pr, \forall l \in SL \tag{8a}$$

$$-A \times (1 - Q1(l)) + 1 \le \sum_{pr \in Pr} Qpr(l) \le A \times Q1(l); \qquad \forall l \in SL \tag{9a}$$

$$-A \times (1 - Q2(l)) + 1 \le \sum_{pr \in Pr} Qpr(l) - 1 \le A \times Q2(l); \qquad \forall l \in SL \tag{10a}$$

$$(Q1(l) - Q2(l)) + Qpr(l) - 1 \leq NQpr(l); \quad \forall l \in SL \tag{11a}$$

Similar constraints hold for each substrate node:

$$-A \times (1 - Qpr(n)) + 1 \leq \sum_{vl \in VL(pr)} \sum_{l \in Inout(n) \cap EL} Xvl, l \leq A \times Qpr(n);$$
$$\forall pr \in Pr, \forall n \in SN \tag{8b}$$

$$-A \times (1 - Q1(n)) + 1 \leq \sum_{pr \in Pr} Qpr(n) \leq A \times Q1(n); \quad \forall n \in SN \tag{9b}$$

$$-A \times (1 - Q2(n)) + 1 \leq \sum_{pr \in Pr} Qpr(n) - 1 \leq A \times Q2(n); \quad \forall n \in SN \tag{10b}$$

$$(Q1(n) - Q2(n)) + Qpr(n) - 1 \leq NQpr(n); \quad \forall n \in SN \tag{11b}$$

The objective function to be minimized is the total amount of substrate resources required. First, the traffic volume on a substrate link due to virtual link *vl* is denoted by TL_{vl}, and the traffic volume on a substrate node due to one of two service components for virtual link *vl* is denoted by TN_{vl}. When the requested virtual network shares no bandwidth of substrate link *l* with the existing virtual networks, the required bandwidth of substrate link *l* due to the aggregated traffic volume T_l from the requested virtual network is given by expressions (2) and (3) as follows:

$$Bl = \overline{Tl} + kl\sigma(Tl)$$
$$= \sum_{vl \in VL} \overline{TLvl} Xvl, l + \frac{\sum_{vl \in VL} kvl \overline{TLvl} Xvl, l}{\sum_{vl \in VL} \overline{TLvl} Xvl, l} \sqrt{\sum_{vl \in VL} \sigma^2(TLvl) Xvl, l}$$

When the requested virtual network shares no capacity of substrate node *n* with the existing virtual networks, the required capacity of substrate node *n* due to the aggregated traffic volume T_n from the requested virtual network is given by expressions (2) and (3):

$$Bn = \overline{Tn} + kn\sigma(Tn)$$
$$= \sum_{vl \in VL} \overline{TNvl} \left(\sum\nolimits_{l \in Inout(n) \cap EL} Xvl, l \right)$$
$$+ \frac{\sum_{vl \in VL} kvl \overline{TNvl} \left(\sum\nolimits_{l \in Inout(n) \cap EL} Xvl, l \right)}{\sum_{vl \in VL} \overline{TNvl} \left(\sum\nolimits_{l \in Inout(n) \cap EL} Xvl, l \right)}$$
$$\times \sqrt{\sum_{vl \in VL} \sigma^2(TNvl) \left(\sum\nolimits_{l \in Inout(n) \cap EL} Xvl, l \right)}$$

When existing virtual network *vnet* shares no bandwidth of substrate link *l* with the other virtual networks, the required bandwidth due to the traffic volume $T_{vnet,l}$ from *vnet* is denoted by $B_{vnet,l}$. When substrate link bandwidth sharing among multiple virtual networks is achieved according to the policy explained in Section 3.3, the required bandwidth of substrate link *l* due to the total traffic volume from the existing and requested virtual networks can be formulated using expressions (2) and (4) as follows:

$$TBl = \sum_{vnet \in VNet} Bvnet, lEQ2vnet(l) + BlQ2(l)$$
$$+ \sum_{pr \in Pr} \left\{ \sum_{vnet \in VNet} \overline{Tvnet, l} ENQpr(l) + \overline{Tl} NQpr(l) \right.$$
$$\left. + kpr, l \sqrt{\sum_{vnet \in VNet} \sigma^2(Tvnet, l) ENQpr(l) + \sigma^2(Tl) NQpr(l)} \right\}$$

In the above expression, the first term on the right side indicates the required bandwidth for the existing virtual networks that share no bandwidth with the other virtual networks. The second term indicates the required bandwidth for the requested virtual network when the requested virtual network shares no bandwidth with the existing virtual networks. The third term indicates the required bandwidth for the existing and requested virtual networks that share substrate link bandwidth amongst each other. The third term sums the required bandwidth for the existing and requested virtual networks where only virtual links belonging to priority class *pr* traverse the substrate link *l*. The value of $k_{pr,l}$ is formulated using expression (3) as follows:

$$kpr, l = \frac{\sum_{vnet \in VNet} kvnet, l\overline{Tvnet, l} ENQpr(l) + kl\overline{Tl} NQpr(l)}{\sum_{vnet \in VNet} \overline{Tvnet, l} ENQpr(l) + \overline{Tl} NQpr(l)}$$

Likewise, the capacity required due to the traffic volume $T_{vnet,n}$ from existing virtual network *vnet* is denoted by $B_{vnet,n}$ when *vnet* shares no capacity of substrate node *n* with the other virtual networks. The required capacity of substrate node *n* due to the total traffic volume from the existing and requested virtual networks can be formulated using expressions (2) and (4) as follows, when the substrate node capacity share among multiple virtual networks is achieved according to the policy described in Section 3.3:

$$TBn = \sum_{vnet \in VNet} Bvnet, nEQ2vnet(n) + BnQ2(n)$$
$$+ \sum_{pr \in Pr} \left\{ \sum_{vnet \in VNet} \overline{Tvnet, n} ENQpr(n) + \overline{Tn} NQpr(n) \right.$$
$$\left. + kpr, n \sqrt{\sum_{vnet \in VNet} \sigma^2(Tvnet, n) ENQpr(n) + \sigma^2(Tn) NQpr(n)} \right\}$$

The value of $k_{pr,n}$ is formulated using expression (3) as follows:

$$kpr, n = \frac{\sum_{vnet \in VNet} kvnet, n\overline{Tvnet, n} ENQpr(n) + kn\overline{Tn} NQpr(n)}{\sum_{vnet \in VNet} \overline{Tvnet, n} ENQpr(n) + \overline{Tn} NQpr(n)}$$

If the maximum bandwidth of each substrate link *l* and maximum capacity of each substrate node *n* are restricted to MB_l and MB_n, additional constraints are necessary for the integer programming model as follows:

$$TBl \leq MBl; \quad \forall l \in SL \tag{12a}$$

$$TBn \leq MBn; \quad \forall n \in SN \tag{12b}$$

Finally, the total amount of substrate resources required can be formulated as follows:

$$Obj1 = W1 \times \sum_{l \in SL} TBl + W2 \times \sum_{n \in SN} TBn \tag{13}$$

In the above expression, the notations *W1* and *W2* indicate weights for the substrate link resource and substrate node resource. The above objective function is a nonlinear complex function and rarely converges to the minimum value.

In this paper, a simpler linear objective function is introduced to solve the VNE problem using an integer linear programming (ILP) model. Two kinds of variables $NNQ_{pr}(l)$ and $NNQ_{pr}(n)$ shown in the bottom of Table 2 are further added to the model. From the definition of the variables, the following constraints hold for each substrate link and substrate node:

$$\forall pr \in Pr$$
$$ENQpr, vnet(l) \leq NNQpr(l); \quad \forall vnet \in VNet, \quad \forall l \in SL \tag{14a}$$
$$(Q1(l) - Q2(l)) + Qpr(l) - 1 \leq NNQpr(l); \quad \forall l \in SL$$

$$\forall pr \in Pr$$
$$ENQpr, vnet(n) \leq NNQpr(n); \quad \forall vnet \in VNet, \quad \forall n \in SN \tag{14b}$$
$$(Q1(n) - Q2(n)) + Qpr(n) - 1 \leq NNQpr(n); \quad \forall n \in SN$$

The linear objective function simply considers the substrate link bandwidth shared among multiple virtual links of different priorities within a virtual network and the substrate link bandwidth shared among multiple virtual networks as a unit of substrate link bandwidth. Likewise, the substrate node capacity shared among multiple service components having different priorities within a virtual node and the substrate node capacity shared among multiple virtual nodes are regarded as a unit of substrate node capacity. This is equivalent to the region surrounded by the solid lines in Fig. 2 always being counted as one unit of substrate resource. For example, the amount of substrate resources is 2.0 in Fig. 2(a), and 1.0 in Fig. 2(b) and (c). Then, the objective function to represent the total amount of substrate resources required is given as follows:

$$Obj2 = W1 \times \sum_{l \in SL} \left\{ \sum_{vnet \in VNet} EQ2vnet(l) + Q2(l) + \sum_{pr \in Pr} NNQpr(l) \right\} + W2 \times \sum_{n \in SN} \left\{ \sum_{vnet \in VNet} EQ2vnet(n) + Q2(n) + \sum_{pr \in Pr} NNQpr(n) \right\}; \quad (15)$$

Multiple embedded virtual networks utilize each unit of substrate link bandwidth and substrate node capacity in an isolated manner. Thus, the above objective function also reflects the virtual network scheduling cost, i.e., the total number of queues for scheduling the bandwidth utilization of each substrate link and the capacity utilization of each substrate node by multiple embedded virtual networks. For example, the weighted fair queuing mechanism enables the above virtual network scheduling in each substrate link and substrate node [26].

4. Heuristic virtual network embedding method

Although the considered virtual network embedding (VNE) problem can be formulated using the integer linear programming (ILP) model with the objective function (15), solving the ILP model still requires a large computational effort and it is only applicable to small-scale networks. Thus, this section proposes a heuristic method for the considered VNE problem.

4.1. Procedure for virtual network embedding

A novel heuristic VNE method based on the greedy approach is proposed. The proposed VNE method prioritizes the virtual link assignment rather than the virtual node assignment since substrate resource sharing among multiple virtual networks is more likely to occur on the substrate link bandwidth. The proposed method successively computes the optimum substrate path for each virtual link in the requested virtual network, using the conventional minimum-cost route algorithm [27]. The proposed method adjusts the costs of the substrate links and substrate nodes every time it computes the optimum substrate path for the considered virtual link. The cost of a substrate link reflects the increase in the required substrate link bandwidth when the considered virtual link traverses the substrate link. Likewise, the cost of a substrate node reflects the increase in the required substrate node capacity when the virtual node terminating the considered virtual link is assigned to the substrate node. The optimum substrate path minimizes the increase in the required amount of substrate link bandwidth and substrate node capacity when the considered virtual link is assigned to it.

Fig. 4 provides an outline of the VNE procedure for the proposed heuristic method. A list of virtual links comprising the requested virtual network is configured in step 1 and each virtual link is sequentially assigned to the optimum substrate path in steps 2 through 17. First, a virtual link is randomly selected from the list in step 3. The optimum substrate path for the selected virtual link is computed in steps 4 through 15. In step 5, the costs of all the substrate links are calculated according to the states of the substrate links. The state of a substrate link indicates how many virtual links belonging to each priority class have already been assigned to the substrate link in each of the existing virtual networks and the requested virtual network. A list of pairs of candidate substrate nodes for the two virtual nodes terminating the selected virtual link is configured in step 6. Each virtual node can be assigned to its central substrate node or one of the substrate nodes adjacent to the central substrate node. In step 6, the substrate nodes to which other virtual nodes are already assigned are excluded from the candidate substrate nodes. If the considered virtual node has already been assigned to a substrate node, the candidate substrate node is fixed at the substrate node. Two substrate nodes forming a pair of candidate substrate nodes must be different from each other.

Procedure: Heuristic Virtual Network Embedding

```
1.  Configure a list VLlist of virtual links included in the requested
    virtual network.
2.  while VLlist≠∅ do
3.    Select and remove a virtual link vl from VLlist.
4.    mincost ← ∞.
5.    Calculate the costs of all the substrate links based on their states.
6.    Configure a list SNlist of pairs of candidate substrate nodes for two
      virtual nodes vn1 and vn2 terminating vl.
7.    while SNlist≠∅ do
8.      Select and remove a pair of substrate nodes sn1 and sn2 from
        SNlist.
9.      Compute the minimum-cost substrate path sp between sn1 and
        sn2.
10.     Calculate the costs of sn1 and sn2 based on their states.
11.     Calculate the total cost cost of substrate path sp and pair of
        substrate nodes sn1 and sn2.
12.     if cost < mincost then
13.       optsp ← sp, optsn1 ← sn1, optsn2 ← sn2.
14.     end if
15.   end while
16.   Assign the virtual link vl to the substrate path optsp and the pair of
      virtual nodes vn1 and vn2 to the pair of substrate nodes optsn1 and
      optsn2.
17. end while
```

Fig. 4. Outline of virtual network embedding procedure.

The optimum substrate path for the considered virtual link is computed in steps 7 through 15. First, a pair of candidate substrate nodes is randomly selected from the list in step 8. In step 9, the minimum-cost substrate path between the selected pair of candidate substrate nodes is computed. The cost of the selected pair of candidate substrate nodes is calculated according to the states of the candidate substrate nodes in step 10. The state of a substrate node indicates the number of service components belonging to each priority class that have already been assigned to the substrate node in each of the existing virtual networks. The total cost of the minimum-cost substrate path and the selected pair of candidate substrate nodes is calculated in step 11. The cost of the minimum-cost substrate path and the cost of the pair of candidate substrate nodes may be summed using the weights for the substrate link resource and the substrate node resource, respectively. The combination of the minimum-cost substrate path and pair of candidate substrate nodes that achieves the least total cost is memorized as the optimum substrate path in steps 12 through 14. Finally, the considered virtual link and the two virtual nodes at both ends are assigned to the optimum substrate path in step 16.

Since the computational complexity of the minimum-cost route algorithm is given by O $((|SL|+|SN|) \log |SN|)$ [26], the computational complexity of the proposed heuristic method is given by O $(|VL|\ (d+1)^2\ (|SL|+|SN|) \log |SN|)$ in the worst case and a polynomial order of the scales of the requested virtual network and substrate network. Here, the symbols $|SL|$ and $|SN|$ denote the total numbers of substrate links and substrate nodes, respectively. The symbol $|VL|$ indicates the number of virtual links in the requested virtual network and the symbol d represents the average node degree in the substrate network.

4.2. Calculation of substrate resource cost

Based on the greedy strategy, the cost of a substrate resource is correlated with the increase in the required amount of substrate resources due to assignment of the considered virtual link and virtual node. The costs of all the substrate links are calculated according to their states in step 5 of the VNE procedure shown in Fig. 4. The cost of each substrate link corresponds to the additional bandwidth resulting from the assignment of the considered virtual link. The cost of the substrate link is set to a sufficiently large value if the required bandwidth exceeds the given maximum bandwidth due to the assignment of the considered virtual link. Similarly, the costs of the selected pair of candidate substrate nodes are calculated depending on their states in step 10 of the VNE procedure. The cost of each candidate substrate node indicates the additional capacity required to assign one of the two virtual nodes terminating the considered virtual link. The cost of the substrate node is set to a sufficiently large value when the capacity required exceeds the given maximum capacity due to the assignment of the virtual node. In contrast, the cost of the candidate substrate node is set to zero when the virtual node has already been assigned to the candidate substrate node.

Fig. 5 shows how the substrate link cost is calculated for three different cases. In Fig. 5, the meanings of each region and each line depicting the boundary of two regions are identical to those in Fig. 2. The total number of virtual links in the existing virtual networks that have shared the substrate link bandwidth with those in the requested virtual network is denoted by the notation $m0$. Furthermore, the number of virtual links in the requested virtual network that have already been assigned to the considered substrate link is denoted by the notation $m1$. The first case shown in Fig. 5(a) corresponds to the state where only virtual links belonging to the same priority class as the considered virtual link have been assigned to the substrate link in the requested virtual network. In this case, the requested virtual network can continue sharing the substrate link bandwidth with other virtual networks even if the considered virtual link is assigned to the substrate link. The considered virtual link can share the substrate link bandwidth with virtual links 1 through $m0$ in the existing virtual networks and virtual links $m0+1$ through $m0+m1$ in the requested virtual network. When the traffic volume due to the existing virtual link i is indicated by the notation T_i and the traffic volume due to the considered virtual link is indicated by the notation T, the substrate link cost (C) is calculated using expressions (2) and (3) as follows:

$$C = \overline{T} + \left(\frac{k\overline{T} + \sum_{i=1}^{m0+m1} k\overline{Ti}}{\overline{T} + \sum_{i=1}^{m0+m1} \overline{Ti}} \right) \sqrt{\sigma^2(T) + \sum_{i=1}^{m0+m1} \sigma^2(Ti)} - \left(\frac{\sum_{i=1}^{m0+m1} k\overline{Ti}}{\sum_{i=1}^{m0+m1} \overline{Ti}} \right) \sqrt{\sum_{i=1}^{m0+m1} \sigma^2(Ti)} \quad (16)$$

In the above expression, the first term indicates the increase in the average part of the required substrate link bandwidth. The second/third terms correspond to the deviation part of the shared

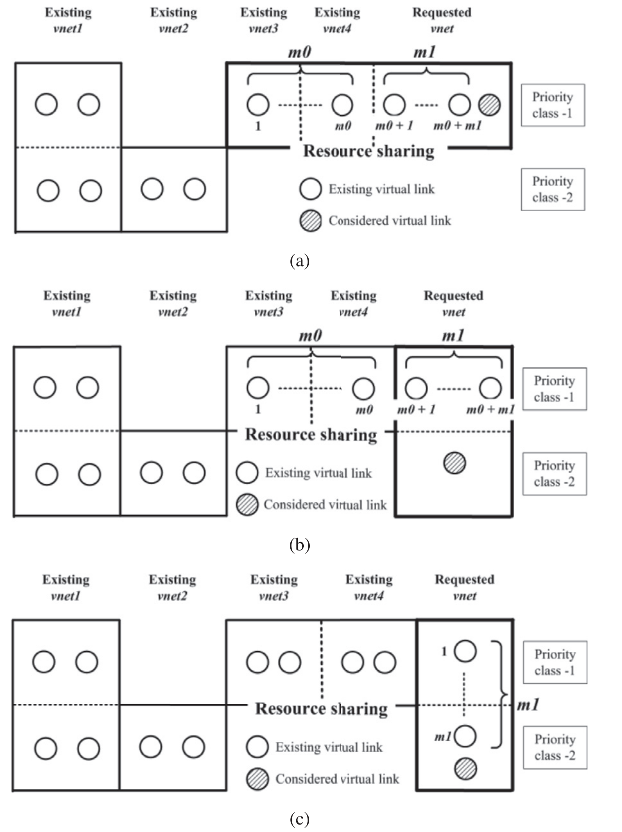


Fig. 5. Three cases for calculating substrate link cost. (a) Case where substrate link bandwidth sharing continues among the existing and requested virtual networks. (b) Case where substrate link bandwidth sharing becomes impossible among the existing and requested virtual networks. (c) Case where substrate link bandwidth sharing is continuously impossible among the existing and requested virtual networks.

link bandwidth after/before the considered virtual link is assigned to the substrate link.

The second case shown in Fig. 5(b) corresponds to the state where only virtual links in an identical priority class, which is different from the priority class for the considered virtual link, have been assigned to the substrate link in the requested virtual network. Once the considered virtual link is assigned to the substrate link, it becomes impossible for the requested virtual network to share the substrate link bandwidth with other virtual networks. Virtual links 1 through $m0$ in the existing virtual networks and virtual links $m0+1$ through $m0+m1$ in the requested virtual network can continue sharing the substrate link bandwidth amongst each other unless the considered virtual link is assigned to the substrate link. However, the considered virtual link can share the substrate link bandwidth only with virtual links $m0+1$ through $m0+m1$ in the requested virtual network if it is assigned to the substrate link. Thus, the substrate link cost (C) in this case is given by the following expression:

$$C = \overline{T} + \left(\frac{\sum_{i=1}^{m0} k\overline{Ti}}{\sum_{i=1}^{m0} \overline{Ti}} \right) \sqrt{\sum_{i=1}^{m0} \sigma^2(Ti)} + \left(\frac{k\overline{T} + \sum_{i=m0+1}^{m0+m1} k\overline{Ti}}{\overline{T} + \sum_{i=m0+1}^{m0+m1} \overline{Ti}} \right) \sqrt{\sigma^2(T) + \sum_{i=m0+1}^{m0+m1} \sigma^2(Ti)} - \left(\frac{\sum_{i=1}^{m0+m1} k\overline{Ti}}{\sum_{i=1}^{m0+m1} \overline{Ti}} \right) \sqrt{\sum_{i=1}^{m0+m1} \sigma^2(Ti)} \quad (17)$$

In the above expression, the first term indicates the increase in the average part of the required substrate link bandwidth. The second term corresponds to the deviation part of the substrate link bandwidth required for the existing virtual networks that have shared the substrate link bandwidth with the requested virtual network prior to the assignment of the considered virtual link. The third term corresponds to the deviation part of the substrate link bandwidth required for the requested virtual network. The last term indicates the deviation part of the substrate link bandwidth that the existing and requested virtual networks have shared amongst each other prior to the assignment of the considered virtual link.

The third case shown in Fig. 5(c) corresponds to the state where multiple virtual links in different priority classes have already been assigned to the substrate link in the requested virtual network. In this case, it is continuously impossible for the requested virtual network to share the substrate link bandwidth with any of the other virtual networks, even if the considered virtual link is assigned. The considered virtual link is able to share the substrate link bandwidth only with virtual links *1* through *m1* in the requested virtual network when it is assigned to the substrate link. In this case, the substrate link cost (C) can be calculated as follows:

$$C = \overline{T} + \left(\frac{k\overline{T} + \sum_{i=1}^{m1} ki\overline{Ti}}{\overline{T} + \sum_{i=1}^{m1} \overline{Ti}} \right) \sqrt{\sigma^2 (T) + \sum_{i=1}^{m1} \sigma^2 (Ti)} - \left(\frac{\sum_{i=1}^{m1} ki\overline{Ti}}{\sum_{i=1}^{m1} \overline{Ti}} \right) \sqrt{\sum_{i=1}^{m1} \sigma^2 (Ti)} \quad (18)$$

In the above expression, the first term indicates the increase in the average part of the required substrate link bandwidth. The second/third terms indicate the deviation part of the substrate link bandwidth required for the requested virtual network after/before the considered virtual link is assigned. If substrate link bandwidth sharing is prohibited among multiple virtual networks and only permitted within the identical virtual network, the substrate link cost is always calculated using the above expression (18). When the substrate link bandwidth is shared among no virtual links, the substrate link cost is simply given by the required bandwidth for the considered virtual link.

Fig. 6 shows how the substrate node cost is calculated for two different cases. Multiple service components may be executed on the considered virtual node. The first case shown in Fig. 6(a) indicates the case where all the service components executed on the considered virtual node belong to an identical priority class. In this case, the considered virtual node can share the substrate node capacity with other existing virtual nodes where only the service components in the same priority class are executed. In Fig. 6(a), service components $m0+1$ through $m0+m1$ executed on the considered virtual node can share the substrate node capacity with service components *1* through *m0* executed on the existing virtual nodes. When the traffic volume due to the service component i is denoted by the notation T_i, the substrate node cost (C) is calculated using expressions (2) and (3) as follows:

$$C = \sum_{i=m0+1}^{m0+m1} \overline{Ti} + \left(\frac{\sum_{i=1}^{m0+m1} ki\overline{Ti}}{\sum_{i=1}^{m0+m1} \overline{Ti}} \right) \sqrt{\sum_{i=1}^{m0+m1} \sigma^2 (Ti)} - \left(\frac{\sum_{i=1}^{m0} ki\overline{Ti}}{\sum_{i=1}^{m0} \overline{Ti}} \right) \sqrt{\sum_{i=1}^{m0} \sigma^2 (Ti)} \quad (19)$$

In the above expression, the first term indicates the increase in the average part of the required substrate node capacity. The second/third terms correspond to the deviation part of the substrate node capacity shared after/before the considered virtual node is assigned to the substrate node.

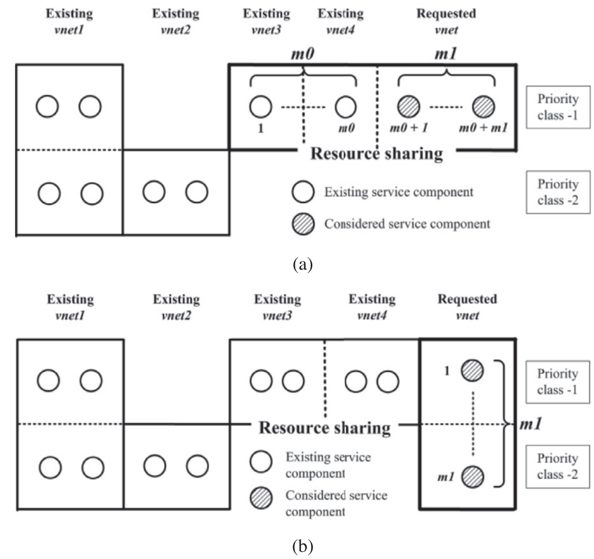


Fig. 6. Two cases for calculating substrate node cost. (a) Case where the considered virtual node can share substrate node capacity with other existing virtual nodes. (b) Case where the considered virtual node cannot share substrate node capacity with the existing virtual nodes.

The second case in Fig. 6(b) indicates where the considered virtual node executes multiple service components belonging to different priority classes. In this case, the considered virtual node cannot share the substrate node capacity with the other virtual nodes even if it is assigned to the substrate node. In Fig. 6(b), service components *1* through *m1* executed on the considered virtual node cannot share the substrate node capacity with any of the existing service components. Thus, the substrate node cost (C) can be calculated as follows:

$$C = \sum_{i=1}^{m1} \overline{Ti} + \left(\frac{\sum_{i=1}^{m1} ki\overline{Ti}}{\sum_{i=1}^{m1} \overline{Ti}} \right) \sqrt{\sum_{i=1}^{m1} \sigma^2 (Ti)} \quad (20)$$

In the above expression, the first and second terms indicate the average and deviation parts of the required substrate node capacity for assigning the considered virtual node. If substrate node capacity sharing is prohibited among multiple virtual nodes and only permitted within each virtual node, the substrate node cost is always calculated using the above expression (20). When the substrate node capacity is not shared among any of the service components even within the same virtual node, the substrate node cost is simply given by the total sum of the required substrate node capacity for each service component in the considered virtual node.

5. Evaluation of virtual network embedding

This section verifies the effectiveness of the proposed virtual network embedding (VNE) method using computer simulation. The total amount of substrate resources required is evaluated to estimate the effect of substrate resource sharing among multiple priority classes and multiple virtual networks. The required amounts of substrate resources in the following three cases are evaluated assuming that the bandwidth in all the substrate links and capacity in all the substrate nodes are unlimited. If the amounts of those substrate resources are restricted, the loss rate of the VNE requests increases due to the increase in the required amount of substrate resources estimated in this section.

Case-1: No virtual link or service component shares the substrate resources with other virtual links or service components

Table 4
Simulation settings.

Setting item	Setting value
Topology of substrate network	Random (tree) network
Total number of substrate nodes (|*SN*|)	20, 100, 150, 201
Total number of substrate links (|*SL*|)	30, 150, 200, 250
Distribution of number of virtual nodes	Binomial distribution
Average number of virtual nodes (|*VN*|)	6, 9,12
Probability that a virtual link exists between a pair of virtual nodes (*P*)	0.25, 0.50. 0.75
Number of priority classes (|*Pr*|)	2, 4 (Each class specified evenly.)
Arrival of virtual network requests	Poisson process
Average arrival rate of virtual network requests	Variable parameter
Lifetime of virtual networks	Negative exponential distribution
Average lifetime of virtual networks	1.0
Average traffic volume in each virtual link and service component ($\overline{T}$)	0.5
Standard deviation of traffic volume for each virtual link and service component ($\sigma(T)$)	Constant
Deviation part of substrate resource required for each virtual link and service component belonging to priority class *pr* ($k_{pr}\sigma(T)$)	0.8, 1.2 when |*Pr*|=2, 0.7, 0.9, 1.1, 1.3 when |*Pr*|=4
Weight for the substrate link bandwidth (*W1*)	1.0
Weight for the substrate node capacity (*W2*)	1.0
Transition state interval in one simulation	5000 requests
Stationary state interval in one simulation	10,000 requests

even if they comprise the same virtual network. The required amount of substrate resources can be calculated from the VNE result using just expression (4).

Case-2: Each virtual link or service component can share the substrate resource with the other virtual links or service components within the same virtual network even if they belong to different priority classes. The required amount of substrate resources can be calculated from the VNE result by applying expressions (2) and (3) within each virtual network and expression (4) among the different virtual networks.

Case-3: Each virtual link or service component can share the substrate resource with other virtual links or service components according to the policy illustrated in Section 3.3. The required amount of substrate resources can be calculated from the VNE result according to expression (13). Furthermore, the virtual network scheduling cost in this case can be calculated from the VNE result using expression (15).

In comparison with *Case-1*, substrate resource sharing is achieved among multiple priority classes within each virtual network in *Case-2*. In comparison with *Case-2*, substrate resource sharing can be also achieved between multiple virtual networks in *Case-3*.

5.1. Simulation settings

The simulation experiments aim to assess the effect of substrate resource sharing among multiple priority classes and multiple virtual networks. Table 4 summarizes the settings for the simulation experiments. Random networks are considered to be the topology of the substrate networks [28]. In the random networks, a link is sequentially configured between each pair of nodes at the same probability. When random tree networks are considered to be the topology of the substrate networks, a substrate link is sequentially configured between each pair of substrate nodes at the same probability under the condition that the addition of the new substrate link can avoid loop occurrence. Generation of the random networks is repeated until a connected graph is obtained as the topology of the substrate network.

The number of virtual nodes in each requested virtual network follows a binomial distribution. The number of virtual nodes is more than one and less than or equal to the total number of substrate nodes. This means that the probability of the number of virtual nodes being vn (>1) can be given by the following expression:

$$P(vn) = P'(vn) / \sum_{vn'=2}^{|SN|} P'(vn')$$
$$P'(vn) = {}_{|SN|}C_{vn} (|VN|/|SN|)^{vn} (1.0 - |VN|/|SN|)^{|SN|-vn} \tag{21}$$

In the above expression, the notations |*SN*| and |*VN*| indicate the total number of substrate nodes and the average number of virtual nodes in each requested virtual network. A virtual link connects a pair of virtual nodes at a given probability *P*. However, each virtual node must be connected with at least one of the other virtual nodes using a virtual link. This means that the topology of the virtual networks is given by the connected random graph as with the substrate networks. Each priority class is evenly specified for each virtual link and pair of service components at both sides of the virtual link. The arrival of VNE requests follows the Poisson process and the lifetime of virtual networks follows a negative exponential distribution. Since the average lifetime of virtual networks is set at 1.0, the arrival rate of VNE requests is equal to the average number of virtual networks embedded simultaneously.

For simplicity, the average traffic volume $\overline{T}$ is assumed to be 0.5 in every virtual link and service component, and the standard deviation of the traffic volume σ(T) is also identical in every virtual link and service component. Furthermore, the deviation part of the required amount of substrate resources $k_{pr}\sigma(T)$ is identical in every virtual link and service component belonging to the same priority class *pr*, and is given as shown in Table 4. When the number of priority classes is two, the deviation part is 0.8 in the low priority class and 1.2 in the high priority class. When the number of priority classes is four, the deviation part increases from 0.7 in the lowest priority class to 1.3 in the highest priority class. Then, the amount of substrate resources required for a set of m virtual links or service components that involves m_{pr} virtual links or service components of each priority class *pr* is derived from expressions (2) and (3) as follows, when those m virtual links or service components share the substrate resource with each other:

$$B = 0.5m + \sum_{pr} (kpr\sigma(T) \times \sqrt{mpr}) \tag{22}$$

Meanwhile, the amount of substrate resources required for a set of m virtual links or service components is derived from expression (4), when those m virtual links or service components cannot share the substrate resource:

$$B = 0.5m + \sum_{pr} (kpr\sigma(T) \times mpr) \tag{23}$$

It can be expected from expressions (22) and (23) that the effect of substrate resource sharing strengthens due to the increase in the ratio of virtual links or service components belonging to higher priority classes, although the total required amount of substrate resources increases. However, the following simulation experiments specify the ratio such that the virtual links or service components of each priority class are distributed equally in order to estimate the effect of statistical multiplexing apart from the impact of the ratio.

When the required amount of substrate resources is calculated, both the weights for the substrate link bandwidth *W1* and substrate node capacity *W2* are set at 1.0. In each simulation, the initial interval for the first 5000 virtual network requests is regarded as the transition state. The following interval for 10,000 virtual network requests is regarded as the stationary state and the statistics required are collected from the measurement results during the interval.

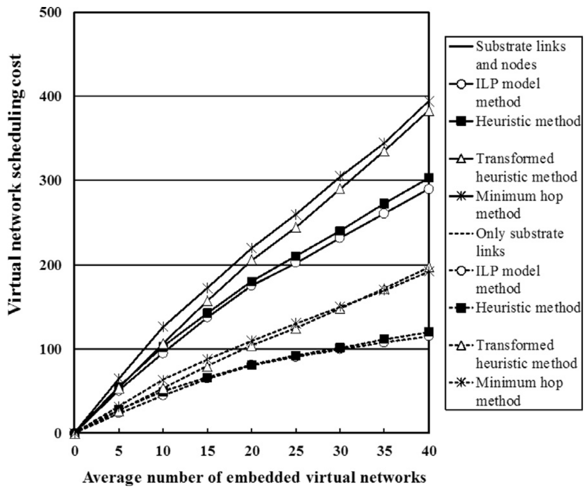


Fig. 7. Virtual network scheduling cost in *Case-3*.

5.2. Comparison of virtual network embedding methods

In this subsection, four methods for VNE are compared. The first method referred to as the “ILP model method” directly solves the integer linear programming (ILP) model formulated in Section 3.4. The objective function is given by expression (15) from the perspective of the convergence to the optimum solution. This means that the “ILP model method” can minimize the virtual network scheduling cost in *Case-3*. The second method referred to as the “Heuristic method” is the one proposed in Section 4, which aims to minimize the required amount of substrate resources in *Case-3*. The third method referred to as the “Transformed heuristic method” aims to minimize the required amount of substrate resources in *Case-2* based on the greedy approach. Although the “Transformed heuristic method” also computes the minimum-cost route for each virtual link repeatedly, the method always calculates the cost of each substrate link and substrate node using expressions (18) and (20), respectively. The last method referred to as the “Minimum hop method” indicates the other heuristic method where each virtual link is successively assigned to the substrate path with the least number of hops. This method aims to minimize the required amount of substrate resources in *Case-1*.

Fig. 7 shows the virtual network scheduling cost in *Case-3* calculated from the VNE result in each of the above four methods. Solid lines indicate the total scheduling cost for all the substrate links and substrate nodes, while, broken lines indicate a part of the scheduling cost for all the substrate links. Fig. 7 shows the virtual network scheduling cost averaged during the stationary state interval in the simulation. The horizontal axis indicates the average number of virtual networks embedded simultaneously. The number of priority classes ($|Pr|$) is two. Ten substrate networks with 20 substrate nodes and 30 substrate links are evaluated by executing the simulation ten times, and the average values derived from those ten substrate networks are plotted in Fig. 7. The above substrate networks represent the largest network scale for which the “IP model method” can obtain the solution within a reasonable time. The average number of virtual nodes is 6 and the probability of virtual link existence is 0.5 in reference to the existing studies [5,12]. As shown in Fig. 7, the “ILP model method” realizes the optimum VNE that minimizes the virtual network scheduling cost in *Case-3*. Since the “Heuristic method” aims to minimize the required amount of substrate resources in *Case-3*, it can accomplish sub-optimum VNE with a virtual network scheduling cost in *Case-3* that is slightly larger than the minimum cost.

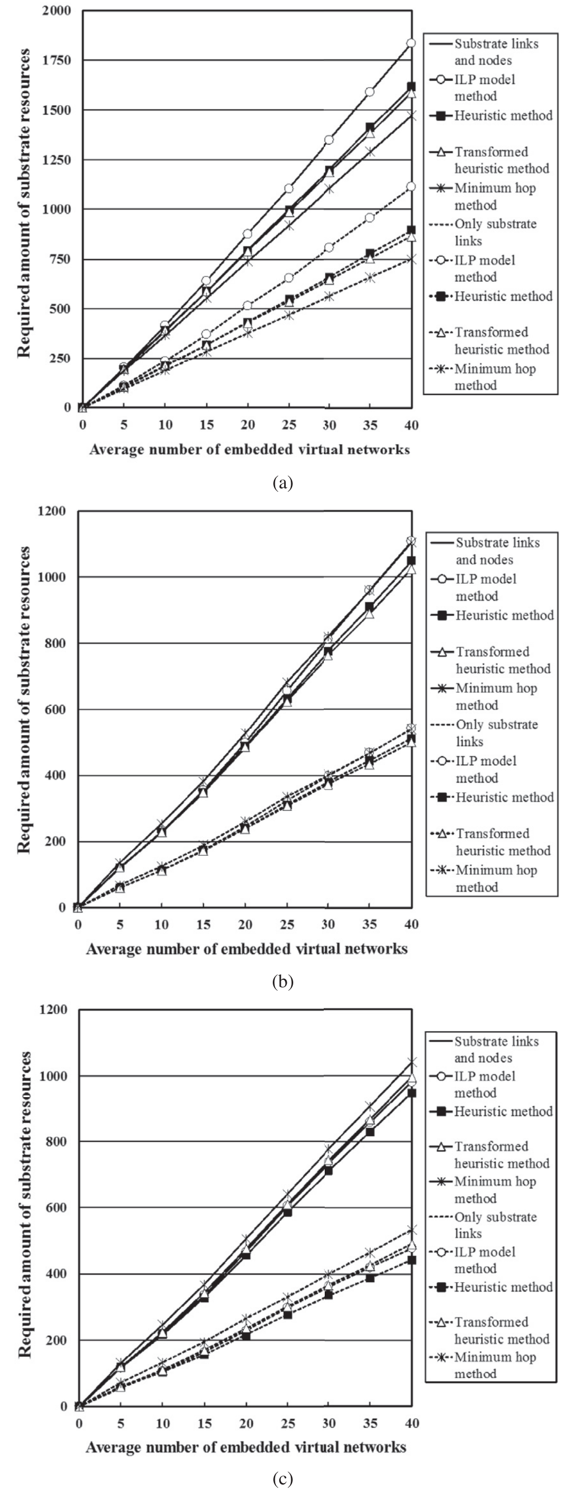


Fig. 8. Comparison of virtual network embedding methods. (a) Required amount of substrate resources in *Case-1*. (b) Required amount of substrate resources in *Case-2*. (c) Required amount of substrate resources in *Case-3*.

Fig. 8 shows the required amount of substrate resources in *Case-1, Case-2*, and *Case-3* calculated from the VNE result in each of the four VNE methods. Solid lines indicate the sum of the required amount of substrate link resources and substrate node resources. Broken lines indicate only the required amount of substrate link resources. Fig. 8 shows the required amount of substrate resources averaged during the stationary state interval in the sim-

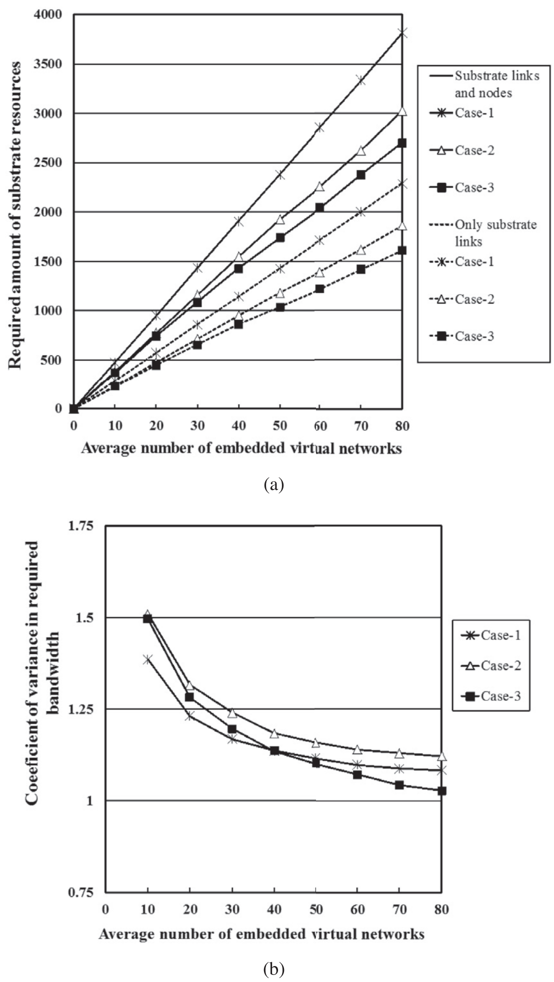


Fig. 9. Effect of substrate resource sharing when the number of priority classes is two. (a) Average amount of substrate resources required. (b) Coefficient of variation in required substrate link bandwidth.

ulation. The number of priority classes (|*Pr*|) is two. Ten substrate networks are identical to those in Fig. 7, and the average values derived from the ten substrate networks are plotted in Fig. 8. The average number of virtual nodes and the probability of virtual link existence are also identical. Fig. 8(a) shows the required amount of substrate resources in *Case-1*. In this case, the VNE that requires the least amount of substrate resources is achieved by the "Minimum hop method". Fig. 8(b) shows the required amount of substrate resources in *Case-2*. In this case, the "Transformed heuristic method" requires the least amount of substrate resources. Fig. 8(c) shows the required amount of substrate resources in *Case-3*. In this case, the best VNE that requires the least amount of substrate resources can be achieved by the "Heuristic method". The best VNE cannot be attained by the "ILP model method" since this method only minimizes the virtual network scheduling cost in *Case-3*. In all three cases, the difference between the required amounts of substrate link resources is primarily responsible for the difference between the total amounts of substrate resources required in each VNE method. Furthermore, the required amount of substrate resources increases approximately in proportion to the average number of virtual networks embedded simultaneously.

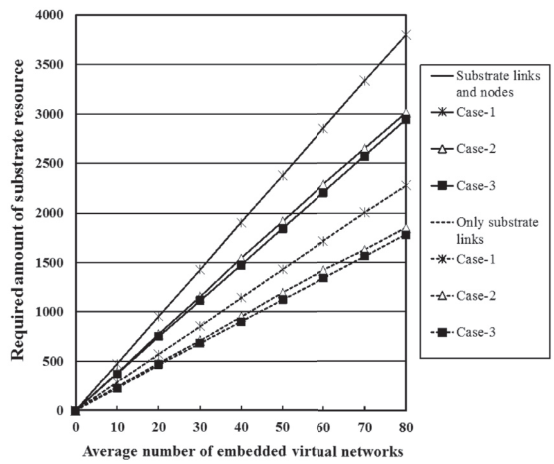
5.3. Effect of substrate resource sharing

Fig. 9 shows the results of the simulation, depicting the effect of substrate resource sharing among multiple priority classes and multiple virtual networks. Fig. 9 evaluates the required amount of substrate resources when the number of priority classes (|*Pr*|) is two. Ten substrate networks with 100 substrate nodes and 150 substrate links are evaluated, and the average values derived from those ten substrate networks are plotted in Fig. 9. The average number of virtual nodes is 6 and the probability of link existence between each pair of virtual nodes is 0.5. The required amounts of substrate resources in *Case-1, Case-2*, and *Case-3* are derived from the VNE results in the methods that aim to minimize the required amount of substrate resources in *Case-1, Case-2*, and *Case-3*. This means that the required amounts of substrate resources in *Case-1, Case-2*, and *Case-3* are calculated from the VNE results in the "Minimum hop method", "Transformed heuristic method", and "Heuristic method", respectively.

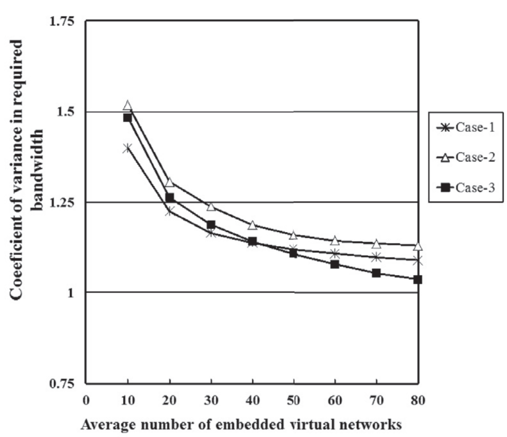
Fig. 9(a) shows the required amount of substrate resources averaged during the stationary state interval in the simulation. From the comparison of *Case-1* and *Case-2*, the required amount of substrate resources can be reduced due to substrate resource sharing among multiple priority classes within each virtual network. From the comparison of *Case-2* and *Case-3*, substrate resource sharing among multiple virtual networks can further reduce the required amount of substrate resources while achieving fair substrate resource sharing according to the policy mentioned in Section 3.3. As shown in Fig. 9(a), substrate resource sharing is primarily effective in reducing the required substrate link bandwidth, which indicates the saving of costly substrate link bandwidth.

Fig. 9(b) shows the coefficient of variation in the required bandwidth for each substrate link, which is averaged during the stationary state interval in the simulation. The coefficient of variation in *Case-2* is greater than that in *Case-1* since substrate resource sharing within each virtual network gathers multiple virtual links in a virtual network into the same substrate link. In contrast, the coefficient of variation in *Case-3* is less than that in *Case-1* when many virtual networks are simultaneously embedded into the substrate network. This is because substrate resource sharing among multiple virtual networks distributes multiple virtual links belonging to the different priority classes to the different substrate links. This property becomes more significant when the average number of embedded virtual networks increases.

Fig. 10 shows the required amount of substrate resources when the number of priority classes (|*Pr*|) is four. The values of parameters pertaining to the substrate networks and the requested virtual networks are identical to those in Fig. 9. The required amounts of substrate resources in *Case-1, Case-2*, and *Case-3* are obtained from the "Minimum hop method", "Transformed heuristic method", and "Heuristic method", respectively. Fig. 10(a) shows the required amount of substrate resources averaged during the stationary state interval. The required amount of substrate resources decreases due to substrate resource sharing among multiple priority classes and multiple virtual networks. Substrate resource sharing is primarily effective in reducing the required substrate link bandwidth. When the number of priority classes increases, less virtual links of the same priority can be gathered into an identical substrate link and virtual links in the different priority classes can hardly be distributed to the different substrate links. Thus, substrate resource sharing among multiple virtual networks becomes less effective due to the increase in the number of priority classes. Fig. 10(b) show the coefficient of variation in the required substrate link bandwidth. The coefficient of variation in *Case-2* is greater than that in *Case-1* and the coefficient of variation in *Case-3* is less than that in *Case-1*. The coefficient of variation in *Case-3* is slightly reduced when the number of priority classes increases. This is because multiple virtual links belonging to the different priority classes are distributed more extensively due to the increase in the number of priority classes.



(a)



(b)

Fig. 10. Effect of substrate resource sharing when the number of priority classes is four. (a) Average amount of substrate resources required. (b) Coefficient of variation in required substrate link bandwidth.

5.4. Influence of topology in virtual and substrate networks

Fig. 11 shows the simulation results that clarify the influence of the topology in requested virtual networks. When the average number of embedded virtual networks is small, the difference between the required amounts of substrate resources in *Case-1, Case-2* and *Case-3* is negligible. Thus, Fig. 11 depicts the ratio of the required substrate link bandwidth in the three cases. In Fig. 11, solid lines indicate the ratio of the required substrate link bandwidth in *Case-3* to that in *Case-2* and represent the effect of substrate link bandwidth sharing among multiple virtual networks. The broken lines indicate the ratio of the required substrate link bandwidth in *Case-2* to that in *Case-1* and represent the effect of substrate link bandwidth sharing among the different priority classes within each virtual network. The required substrate link bandwidth in *Case-1, Case-2*, and *Case-3* is calculated from the VNE results in the “Minimum hop method”, “Transformed heuristic method”, and “Heuristic method”, respectively. Fig. 11 shows the ratios when the average number of virtual nodes ($|VN|$) and the probability of virtual link existence (P) vary. The number of priority classes ($|Pr|$) is two. In the substrate networks, the number of substrate nodes and the number of substrate links are always 100 and 150, respectively. The average values derived from ten substrate networks are plotted in Fig. 11.

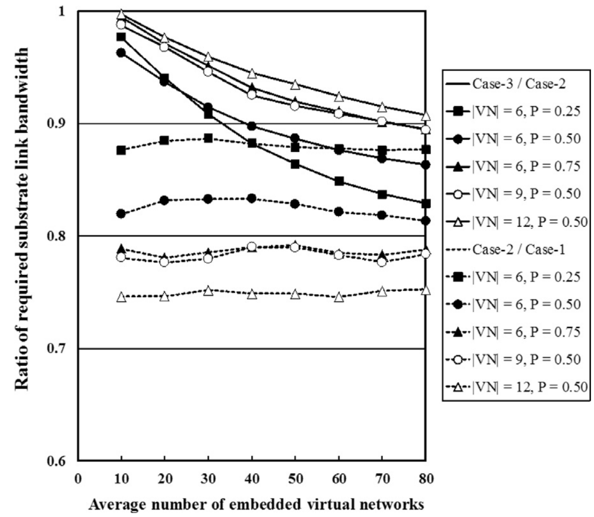


Fig. 11. Ratio of required substrate link bandwidth when topology of requested virtual networks varies.

As shown in Fig. 11, the ratio of the required substrate link bandwidth in *Case-2* to that in *Case-1* can be reduced by the increase in the probability of virtual link existence under the given number of virtual nodes and the increase in the average number of virtual nodes under the given probability of virtual link existence. This means that substrate link bandwidth sharing within each virtual network becomes more effective as the number of virtual links increases in each virtual network. If each virtual network includes more virtual links, the substrate link bandwidth can be shared among more virtual links comprising each of the virtual networks. In contrast, the ratio of the required substrate link bandwidth in *Case-3* to that in *Case-2* can be reduced due to the decrease in the probability of virtual link existence under the given number of virtual nodes and the decrease in the average number of virtual nodes under the given probability of virtual link existence. This means that substrate link bandwidth sharing among multiple virtual networks can be achieved easily when the node degree in the requested virtual networks decreases. If the requested virtual network has a small node degree, it is easy to assign fewer virtual links starting from the same virtual node and belonging to different priority classes to different outgoing substrate links. The effect of substrate link bandwidth sharing among multiple virtual networks becomes more significant as the average number of virtual networks embedded simultaneously increases.

Fig. 12 shows the ratios when the number of substrate nodes ($|SN|$) and number of substrate links ($|SL|$) vary in the substrate networks. The number of priority classes ($|Pr|$) is two. The average number of virtual nodes is 6 and the probability of virtual link existence is 0.5 in the requested virtual networks. The average values derived from ten substrate networks with the same numbers of substrate nodes and substrate links are plotted in Fig. 12. The broken lines show that the substrate networks with a smaller node degree strengthen the effect of substrate link bandwidth sharing within each virtual network. Substrate link bandwidth sharing among multiple virtual links comprising each of the virtual networks can often be achieved when the substrate path candidates for the virtual links are restricted and overlapped as in the random tree topology ($|SN|=201$, $|SL|=200$). The solid lines show that the substrate networks with a larger node degree strengthen the effect of substrate link bandwidth sharing among multiple virtual networks. A higher number of possible substrate paths is desirable in order to distribute multiple virtual links belonging to the different priority classes into the different substrate links. As shown

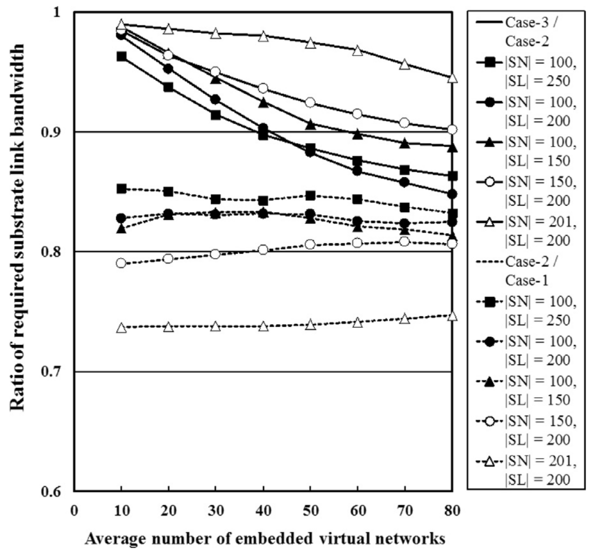


Fig. 12. Ratio of required substrate link bandwidth when topology of substrate networks varies.

in Fig. 12, substrate link bandwidth sharing among multiple virtual networks again becomes more effective as the average number of virtual networks embedded simultaneously increases.

6. Conclusions

This paper formulated a novel virtual network embedding (VNE) problem and proposed a heuristic VNE method to minimize the total amount of substrate resources when each virtual network requires the substrate resources to be shared among its multiple priority classes. The extensive simulations clarified that the proposed VNE method can maximize substrate resource sharing among multiple virtual networks while satisfying the condition for fair substrate resource sharing. This paper quantitatively estimated the effect of substrate resource sharing within each virtual network and among multiple virtual networks. Substrate resource sharing is primarily effective for reducing the required substrate link bandwidth, which means the costly physical link resources can be saved. Substrate resource sharing among multiple virtual networks is significant when the number of priority classes is small. Virtual networks including more virtual links and substrate networks with a larger node degree are good candidates from the perspective of substrate link resource sharing within each virtual network. In contrast, virtual networks with smaller node degrees and substrate networks with a larger node degree are good candidates for substrate link resource sharing among multiple virtual networks. As the number of embedded virtual networks increases, substrate resource sharing among multiple virtual networks becomes more effective for reducing the total required amount of substrate resources.

Acknowledgments

The authors would like to thank Dr. Nakajima, President and CEO, and Dr. Ano, Executive Director of KDDI R&D Laboratories, Inc., for their encouragement throughout the study.

References

[1] J. Turner, D. Taylor, Diversifying the internet, in: Proceedings of IEEE GLOBECOM 2005, vol. 2, December 2005.
[2] N. Feamster, L. Gao, J. Rexford, How to lease the Internet in your spare time, ACM SIGCOMM Comput. Commun. Rev. 37 (January) (2007) 61–64.
[3] N. Choudhury, R. Boutaba, A survey of network virtualization, Comput. Netw. 54 (April (5)) (2010) 862–876.
[4] A. Fischer, J.F. Botero, M.T. Beck, H. de Meer, X. Hesselbach, Virtual network embedding: a survey, IEEE Commn. Surv. Tut. 15 (4) (2013) 1888–1906.
[5] Y. Zhu, M. Ammar, Algorithms for assigning substrate network resources to virtual network components, in: Proceedings of IEEE INFOCOM 2006, April 2006.
[6] X. Cheng, S. Su, Z. Zhang, H. Wang, F. Yang, Y. Luo, J. Wang, Virtual network embedding through topology-aware node ranking, ACM SIGCOMM Comput. Commun. Rev. 41 (April) (2011) 38–47.
[7] S. Zhang, Z. Qian, B. Tang, J. Wu, S. Lu, Opportunistic bandwidth sharing for virtual network mapping, in: *Proceedings of IEEE GLOBECOM 2011*, December, 2011.
[8] S. Zhang, Z. Qian, J. Wu, S. Lu, An opportunistic resource sharing and topology-aware mapping framework for virtual networks, in: Proceedings of IEEE INFOCOM 2012, April 2012.
[9] S. Qing, Q. Qi, J. Wang, T. Xu, J. Liao, Topology-aware virtual network embedding through Bayesian network analysis, in: Proceedings of IEEE GLOBECOM 2012, December 2012, pp. 2621–2627.
[10] L. Gong, Y. Wen, Z. Zhu, T. Lee, Toward profit-seeking virtual network embedding algorithm via global resource capacity, in: Proceedings of IEEE INFOCOM 2014, April 2014.
[11] M. Yu, Y. Yi, J. Raxford, M. Chiang, Rethinking virtual network embedding: substrate support for path splitting and migration, ACM SIGCOMM Comput. Commun. Rev. 38 (April) (2008) 19–29.
[12] M. Chowdhury, M.R. Rahman, R. Boutaba, ViNEYard: virtual network embedding algorithms with coordinated node and link mapping, IEEE/ACM Trans. Netw. 20 (1) (February 2012) 206–219.
[13] N. Butt, N. Choudhury, R. Boutaba, Topolgy-awareness and re-optimization mechanism for virtual network embedding, in: Proceedings of Networking, 2010, pp. 27–39.
[14] Z. Wang, Y. Han, T. Lin, H. Tang, S. Ci, Virtual network embedding by exploiting topological information, in: Proceedings of IEEE GLOBECOM 2012, December 2012, pp. 2603–2608.
[15] R. Riggio, F.D. Pellegrini, E. Salvadori, M. Gerola, R.D. Corin, Progressive virtual topology embedding in OpenFlow networks, in: Proceedings of IFIP/IEEE IM2013 Workshop, 2013, pp. 1123–1128.
[16] S. Zhang, J. Wu, S. Lu, Virtual network embedding with substrate support for parallelization, in: Proceedings of IEEE GLOBECOM 2012, December 2012, pp. 2615–2620.
[17] Q. Hu, Y. Wang, X. Cao, Resolve the virtual network embedding problem: a column generation approach, in: Proceedings of IEEE INFOCOM 2013, April 2013.
[18] L. Gong, Z. Zhu, Virtual optical network embedding (VONE) over elastic optical networks, IEEE/OSA JLT 32 (3) (February 2014) 450–460.
[19] B. Guo, C. Qiao, J. Wang, H. Yu, Y. Zuo, J. Li, Z. Chen, Y. He, Survivable virtual network design and embedding to survive a facility node failure, IEEE/OSA JLT 32 (3) (February 2014) 483–493.
[20] L. Nonde, T.E.H. El-Gorashi, J.M.H. Elmirghani, Energy efficient virtual network embedding for cloud networks, IEEE/OSA JLT 33 (9) (May 2015) 1828–1849.
[21] Network function virtualization (NFV); Terminology for main concepts in NFV, ETSI GS NFV 003 V. 1.1.1 (October 2013).
[22] 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 Netw. Commun. 23 (2) (2012) 109–122.
[23] T. Trinh, H. Esaki, C. Aswakul, Quality of service using careful overbooking for optimal virtual network resource allocation, in: Proceedings of ECTI-CON 2011, May 2011, pp. 296–299.
[24] L. Kleinrock, Queuing Sustems, vol. I-II, John Wiley and Sons, 1976.
[25] N.K. Jaiswal, Priority Queues, Academic Press, 1968.
[26] A. Demers, S. Keshav, S. Shenker, Analysis and simulation of a fair queueing algorithm, ACM SIGCOMM Comput. Commun. Rev. 19 (4) (Sept. 1989) 1–12.
[27] E.W. Dijkstra, A note on two problems in connexion with graphs, Numerische Mathematik 1 (1959) 269–271.
[28] B. Bollobas, Random Graphs, second ed., Cambridge University Press, 2001.

Nagao Ogino received B.E., M.E., and Dr. Eng. degrees from the University of Tokyo, Tokyo, Japan, in 1977, 1979, and 1982, respectively. He joined the Research and Development Laboratories of Kokusai Denshin Denwa Co. Ltd. (currently KDDI Corporation), Tokyo, Japan, in 1982. Since 1982, he has been engaged in research on ATM networks, Intelligent Networks, and telecommunication software engineering. He was a supervisor of the Adaptive Communications Research Laboratories at the ATR Institute, Kyoto, Japan, from 1996 until 2000, where he was engaged in research on multi-agent based adaptive communication systems. He is currently an R&D manager of the Communications Network Planning Laboratory at KDDI R&D Laboratories Inc. and a guest professor at the University of Electro-Communications, Tokyo, Japan. His current interests include traffic engineering and optimization in future networks.

Takeshi Kitahara received the B.E. and M.E. degrees in Electronics, Information and Communications Engineering from Waseda University, Tokyo, Japan, in 2000 and 2002, respectively. He joined KDDI Corporation in 2002. Since 2002, he has been engaged in research and development including practical applications in the field of network planning. From 2009 to 2010, he was a visiting researcher at Helsinki University of Technology (currently Aalto University), Finland. He is currently a senior manager of the Communications Network Planning Laboratory at KDDI R&D Laboratories, Inc. His current interests include traffic engineering and network operations management. He is a member of IEICE.

Shin'ichi Arakawa received M.E. and D.E. degrees in informatics and mathematical science from Osaka University in 2000 and 2003. From August 2000 to March 2006, he was an Assistant Professor with the Graduate School of Economics, Osaka University, Japan. In April 2006, he moved to the Graduate School of Information Science and Technology, Osaka University, Japan. He has been an Associate Professor from October 2011. His research interests include optical networks and complex networks. He is a member of IEEE and IEICE.

Masayuki Murata received the M.E. and D.E. degrees in Information and Computer Science from Osaka University, Japan, in 1984 and 1988, respectively. In April 1984, he joined Tokyo Research Laboratory, IBM Japan, as a Researcher. From September 1987 to January 1989, he was an Assistant Professor with Computation Center, Osaka University. In February 1989, he moved to the Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University. In April 1999, he became a Professor of Cybermedia Center, Osaka University, and is now with Graduate School of Information Science and Technology, Osaka University since April 2004. He has more than four hundred papers of international and domestic journals and conferences. His research interests include network architecture, and performance modeling and evaluation. He is a member of IEEE, ACM and IEICE.