PAPER *Special Section on Next Generation Photonic Network Technologies*

A Distributed Clustering Method for Hierarchical Routing in Large-Scaled Wavelength Routed Networks

Yukinobu FUKUSHIMA[†], ***Student Member***, **Hiroaki HARAI**[††], **Shin'ichi ARAKAWA**[†††], ***and*** **Masayuki MURATA**[†], ***Members***

SUMMARY The scalability of routing protocol has been considered as a key issue in large–scaled wavelength routed networks. Hierarchical routing scales well by yielding enormous reductions in routing table length, but it also increases path length. This increased path length in wavelength–routed networks leads to increased blocking probability because longer paths tend to have less free wavelength channels. However, if the routes assigned to longer paths have greater wavelength resources, we can expect that the blocking probability will not increase. In this paper, we propose a distributed node–clustering method that maximizes the number of lightpaths between nodes. The key idea behind our method is to construct node–clusters that have much greater wavelength resources from the ingress border nodes to the egress border nodes, which increases the wavelength resources on the routes of lightpaths between nodes. We evaluate the blocking probability for lightpath requests and the maximum table length in simulation experiments. We find that the method we propose significantly reduces the table length, while the blocking probability is almost the same as that without clustering.

key words: *WDM, lightpath network, path-vector routing, hierarchical routing, distributed clustering*

1. Introduction

WDM lightpath networks are one of the most promising candidates for the next generation Internet. A lightpath, where signals are handled optically at intermediate nodes, is configured to transport traffic. An optical cross–connect (OXC) switches the wavelengths of each input port to appropriate output ports at each intermediate node. The configuration for lightpaths consists of a route selection phase and a wavelength reservation phase. Route information in the route selection phase is collected via routing protocols such as OSPF [1] or BGP [2]. Then, reservation protocols such as RSVP–TE [3] reserve wavelength resources along the route.

Many researchers have investigated the routing and wavelength reservation protocols for establishing lightpaths in intra–domain networks. Routing and wavelength reservation protocols that target for the inter–domain network have recently been investigated [4-7]. Bernstein *et al.* [4] specified key requirements for inter–domain routing protocols for optical networks. One of these is the "independence of the internal domain control plane mechanism". Routing and wavelength reservation protocols in the inter–domain network are independent of protocols in the intra–domain network. BGP is the only existing protocol that conforms to these requirements and is widely deployed in the current Internet. We can use a BGP that is extended to lightpath networks (e.g., Optical BGP [6]) as the inter–domain routing and wavelength reservation protocol.

Li *et al.* [8] pointed out that BGP lacks scalability of number of routes, which results from the increased number of nodes. This is because the BGP router's memory size limits the routing table size and therefore BGP will not work with a large number of routes. One promising approach to keeping the routing table size scalable is to introduce *hierarchical routing* [9]. The basic idea behind hierarchical routing is to form a set of nodes into a *cluster* to aggregate route information about nodes far from a source node. Each node has complete route information about nodes in the same cluster (i.e., intra–cluster route) and also has aggregated route information about nodes in the other clusters (i.e., inter–cluster route). Therefore, the routing table size is reduced.

Although hierarchical routing reduces the size of the routing table, it generally increases the path length. The main reason is that inter–cluster routes cannot always be the same routes as those in a non–clustered environment. That is, path length is increased when an inter–cluster route with a minimum cluster–hop count differs from the shortest path with a minimum node–hop count (Fig. 1). This increased path length is likely to increase the blocking probability for lightpath requests because the probability of finding wavelengths idle on the path decreases as the path length increases. Therefore, it is important to construct clusters to minimize the blocking probability.

In this paper, we propose a method of clustering in a distributed manner to minimize the blocking probability for lightpath requests. To achieve this, we maximize the number of lightpaths between nodes. The key idea behind our method is to construct the node–

Manuscript received February 9, 2005.
Manuscript revised April 22, 2005.
Final manuscript received 0, 2005.

[†]The authors are with the Graduate School of Information Science and Technology, Osaka University
[††]The author is with the National Institute of Information and Communications Technology
[†††]The author is with the Graduate School of Economics, Osaka University

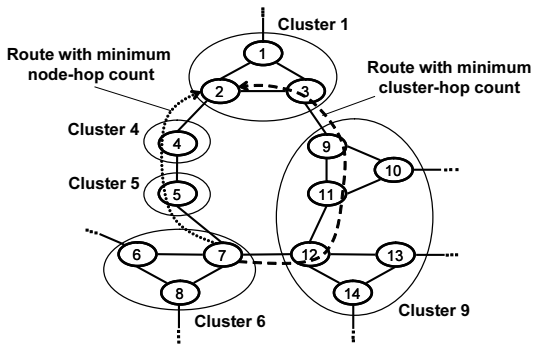


Fig. 1 Route with minimum cluster–hop count and route with minimum node–hop count.

clusters that have many wavelength resources from ingress border nodes to egress border nodes, which increases wavelength resources on the routes of lightpaths. We expect the increased number of available lightpaths would lead to decreased blocking probability. Our method is a distributed clustering algorithm that is suited to large–scaled WDM lightpath networks.

This paper is organized as follows. Section 2 discusses hierarchical routing, node clustering and the conventional clustering problem. In Sec. 3, we propose a distributed method of clustering for WDM lightpath networks. Section 4 presents evaluation results obtained by simulation. Finally, we present our conclusions and the directions of future work in Sec. 5.

2. Hierarchical Routing and Node Clustering

2.1 Network Model

Figure 2 outlines our network model. The network itself consists of nodes and links that correspond to a domain or an Autonomous System (AS) and a set of optical fibers. Note that each node has its own network (i.e., intra–domain WDM lightpath network) but since we focus on the inter–domain WDM lightpath network, the intra–domain lightpath network is represented as a single node. The numbers attached to the links represent the number of fibers on the link in Fig. 2.

When a lightpath is requested, the inter–domain control plane on the gateway of the domain first determines the set of links that the lightpath will traverse (we call the set of links the route) using the route information advertised by the routing protocol, and then reserves wavelength resources along the route using the wavelength reservation protocol. We use a path–vector routing protocol like the BGP for the routing protocol since it meets the requirements of the inter–domain routing protocol in the optical networks [4].

2.2 Hierarchical Clustering

Figure 3 shows an example of hierarchical clustering.

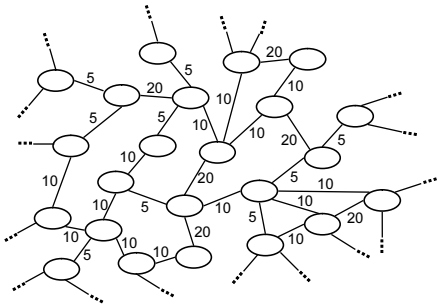


Fig. 2 Network model.

We call a set of nodes a cluster. A node whose adjacent node belongs to another cluster is referred to as a *border node*. A level–x cluster consists of level–$(x-1)$ clusters. The minimum level hierarchy is 1–level clustering, where a level–1 cluster includes all nodes. If the level of clustering is more than 1, this is called multi–level clustering or a multi–level hierarchy.

The maximum cluster size is limited to keep the intra–cluster routing table size within a reasonable size. The inter–cluster routing table size can be huge when there are too many clusters. When this happens, the level of clustering is increased and higher–level clusters are constructed to reduce the size of lower–level inter–cluster tables. Although our approach can be extended to a multi–level hierarchy, we only deal with 2–level hierarchical clustering to simplify explanation.

2.3 Conventional Clustering Problem

Krishnan *et al.* [10] formulated an optimal clustering problem for communications networks. They treated the problem as a graph partitioning problem and called it the *bounded, connected, min–cut* problem. The objective function of the problem is to minimize the sum of the link cost between clusters.

Bounded, connected, min–cut problem

Given:

- An undirected graph $G = (V, E)$ with edge weights $w : E \rightarrow Z_0^+$
- Upper bound on size of clusters $B \in \{1, \ldots, |V|\}$

The optimal clustering is to obtain the set of clusters V_1, V_2, …, V_k, such that

$$minimize \sum_e w(e) \tag{1}$$

where $e \in E$, $e \notin E_i$, $i \in \{1, 2, \ldots, k\}$ $\forall k \in \{2, \ldots, |V|\}$.
Constraints:

- Graph $G_i = (V_i, E_i)$ that represents the intra–cluster–network of cluster V_i is *connected*
- $1 \leq |V_i| \leq B$, $\forall i \in \{1, 2, \ldots, k\}$

There are two characteristics the clustering problem has in communication networks. First, the clusters need to satisfy *bounded, connected* conditions. A

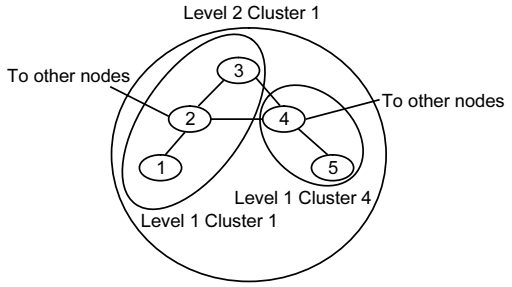


Fig. 3 Example hierarchical clustering.

bounded cluster means the maximum cluster size is bounded by B to keep the intra–cluster routing table within a reasonable size. A *connected* cluster means any two nodes that belong to the same cluster can only reach one another via nodes in that cluster. If the connected condition is not satisfied, two nodes in the same cluster communicate through external clusters. This defeats the purpose of clustering, which is to minimize the storage and exchange of information about external clusters. The second characteristic is that each cluster does not need to be balanced. This is because the construction of balanced clusters does not always result in minimized link costs between clusters.

Krishnan *et al.* [10] proposed a centralized heuristic algorithm to solve the *bounded, connected, min–cut* problem. The heuristic algorithm consists of three steps: (1) generating initial connected clusters, (2) refining clusters by trading nodes, and (3) refining clusters by merging clusters.

The connected clusters in the initial step are generated through recursive bisection. Since the recursive bisection splits clusters, the heuristic algorithm requires the complete information about the entire network topology. This may cause other scalability problems with the memory having to include complete topological information. We therefore propose a clustering algorithm that is implemented in distributed fashion. Our clustering problem and algorithm will be explained in the next section.

3. Node Clustering for Hierarchical Routing in Large–Scaled WDM Lightpath Networks

3.1 Node Clustering in WDM Lightpath Networks

As we discussed in Section 1, clustering may increase the path length. This increase is a serious problem in WDM lightpath networks because the wavelength assigned to a lightpath must be identical along the route (i.e., wavelength continuity constraint). The increased path length generally leads to increased blocking probability for lightpath requests. The routes for lightpaths in hierarchical routing depend on how the clusters are constructed. It is therefore important to construct clusters to minimize the blocking probability for lightpaths.

In this section, we discuss our development of a distributed clustering algorithm that is suited to large–scaled WDM lightpath networks. The requirements for this clustering algorithm are as follows.

1. Keeping the size of routing tables for intra/inter–cluster routing within a certain value
2. Minimizing blocking probability for lightpath requests
3. Constructing clusters in the network with a huge number of nodes

We will explain how these requirements are satisfied with our distributed algorithm after introducing our clustering problem.

To minimize blocking probability in lightpaths, we increase the number of lightpaths available between nodes in WDM lightpath networks. To maximize the number of lightpaths, we first formulate a new clustering problem in WDM lightpath networks that maximizes the number of lightpaths available between nodes. We refer to this problem as the *bounded, connected, max–lightpath* problem. We then propose a distributed clustering algorithm that resolves the *bounded, connected, max–lightpath* problem and satisfies the three requirements.

Bounded, connected, max–lightpath problem
Given:

- $G = (V, E)$ that corresponds to a WDM lightpath network
- Upper bound on size of clusters $B \in \{1, \ldots, |V|\}$

Objective function:

$$maximize \sum_{s=1}^{k} \sum_{i,j \in V_s} F_{ij} + \sum_{s=1}^{k} \sum_{i \in V_s, l \notin V_s} F_{il}, \qquad (2)$$

where V_1, V_2, ..., V_k are constructed clusters. F_{ij} is the number of lightpaths available on the shortest path from node i to node j, where $F_{ii} = 0, (\forall i = 1, \ldots, N)$.
Constraints:

- Graph $G_i = (V_i, E_i)$ that means the intra–network of cluster V_i is *connected*
- $1 \leq |V_i| \leq B$, $\forall i \in \{1, 2, \ldots, k\}$

Let us try to maximize the number of lightpaths available between nodes with the above formulation. The number of lightpaths available between nodes consists of (1) those between nodes in the same cluster and (2) those between nodes in different clusters. The latter changes according to the construction of clusters because route with minimum cluster–hop count, which changes depending on the construction of the clusters, is selected as the route of a lightpath between nodes in different clusters. This route selection follows BGP, where route with minimum AS–hop is selected. We use node–hop/cluster–hop counts as a metric for

intra/inter–cluster route selection. When there are several routes with the same hop counts, we select the route where the minimum number of fibers on links is largest.

The complexity of our *bounded, connected, max–lightpath* problem is open. The complexity of *bounded, connected, min–cut* problem is also open but the related problems such as the bounded, min–k cut problem, where we need to find a subset of edges such that removing them from the graph results in dividing the graph into k subgraphs and the sum of the edge costs in the subset is minimized, are NP-complete [10]. Krishnan *et al.* therefore proposed a heuristic algorithm for the problem. In this paper, we also propose a heuristic algorithm, which satisfies the first and second requirements of a clustering algorithm for large–scale lightpath networks. Our method satisfies the first requirement of "keeping the size of routing tables for intra/inter–cluster routing within a certain value" because the constructed clusters are *bounded* and *connected*. *Bounded* condition limits the number of routes maintained in routing tables. *Connected* condition prevents a node from maintaining intra–cluster routes in different clusters. Our method also satisfies the second requirement of "minimizing the blocking probability for lightpath requests" because it maximizes the number of lightpaths available between nodes. In Sec. 4, we discuss how maximizing available lightpaths results in decreasing the blocking probability for lightpath requests.

For our proposed method to satisfy the third requirement of "constructing clusters in the network with a huge number of nodes", clusters need to be constructed in a distributed fashion. This is because each border node does not need to maintain all the topological information with our method. After we present information maintained by nodes with our method in Sec. 3.2, we will explain our algorithm in Sec. 3.3.

3.2 Information Maintained by Nodes

Figure 4 depicts what information a node and a border node have. All nodes have (1) a *node–to–cluster mapping table* and (2) an *intra–cluster routing table*. In addition, all border nodes have (3) an *inter–cluster routing table*. We will next present the information in each table and when each piece of information is used.

1. Node–to–cluster mapping table:
   This table includes node identifiers and cluster identifiers that include the nodes. We use the minimum node identifier in a cluster as the cluster identifier.
   - When clusters are constructed:
     Each node refers to this table to obtain its cluster identifier, and to find out whether or not it is a border node. Each node can find this out by comparing its cluster identifier with its adjacent nodes' cluster identifiers.
   - When lightpaths are set up:
     Each node refers to this table to obtain the cluster identifier for the destination node.

2. Intra–cluster routing table:
   This table includes the shortest route from a source node to nodes in the same cluster and the minimum number of fibers on links along the route. In the intra–cluster route information to node 2 in Fig. 4, "$1, 2$" is a list of nodes on the route and "$F : 5$" means the minimum number of fibers along the route, which is 5.
   - When clusters are constructed:
     Each border node refers to this table to find out the number of fibers available from it to other border nodes in the same cluster.
   - When lightpaths are set up:
     Each node refers to this table to find out the route to nodes in the same cluster.

3. Inter–cluster routing table:
   This table includes a list of clusters on routes from the source cluster to other clusters and ingress/egress border nodes for each cluster in the list, and the minimum number of fibers on links along the route. In the inter–cluster route information for cluster 7 in Fig. 4, "$(1, 1, 1), (11, 9, 10), (7, 7, -)$" is a list of clusters on the route. Each cluster is expressed as $(ingress\ border\ node\ identifier, cluster\ identifier, egress\ border\ node\ identifier)$. "$F : 5$" means the minimum number of fibers along the route, which is 5.
   - When lightpaths are set up:
     Each border node refers to this table to obtain the route to the destination cluster that includes the destination node.

The inter–cluster routing table includes the ingress/egress border nodes for each cluster. This is because we distinguish the routes that pass through the same clusters but pass through different ingress/egress border nodes. We need to distinguish them because the number of fibers available on a route depends on the ingress/egress border nodes in addition to the clusters a lightpath traverses. Note that a node and/or a border node has only one route information for each destination node/cluster because maintaining multiple routes for a destination leads to increasing routing table size. How to realize a diverse routing, which provides multiple paths that do not share the same nodes or links for increasing reliability, in BGP-based inter–domain routing protocol for optical networks is an important problem as described in [4]. However, this problem is beyond the scope of this paper.

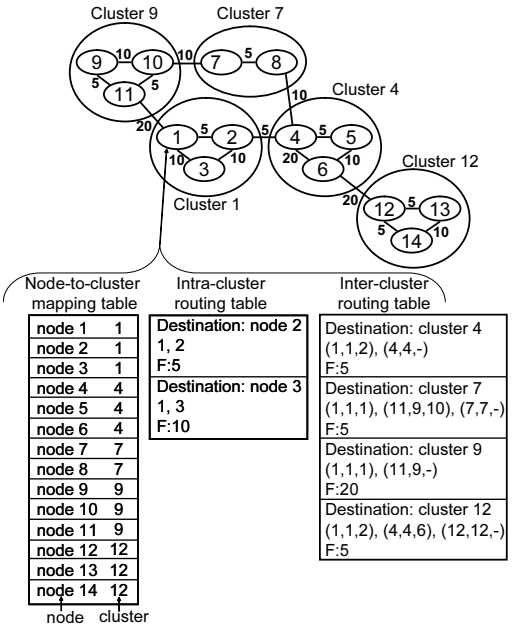


Fig. 4 (1) Node–to–cluster mapping table, (2) inter-cluster routing table, and (3) intra-cluster routing table.

3.3 Distributed Clustering Algorithm for Bounded, Connected, Max–Lightpath Problem

Our algorithm constructs clusters by repeating a *merge* operation. The merge operation makes a cluster merge with an adjacent cluster.

Each cluster performs merge operation with an adjacent cluster so that Eq. (2) is maximized. The first term in Eq. (2), which means the number of lightpaths whose source and destination belong to the same cluster, is constant despite the construction of the clusters. This is because the routes for those lightpaths are always routes with a minimum node–hop count. The second term in Eq. (2), on the other hand, which means the lightpaths whose source and destination belong to different clusters, changes according to the construction of the clusters because their routes have a minimum cluster–hop count. Consequently, it is important to increase F_{il} in the second term.

In order to maximize F_{il}, it is important to prevent lightpaths that traverse several clusters from being routed on links with few fibers. If the links with few fiber are located between clusters, those links do not tend to be selected as routes for lightpaths. This is because there are multiple links between the clusters and the link with the most fibers is selected as the route among them. Thus, we try to locate links with more fibers in clusters, and to locate links with few fibers between clusters. To achieve this, we use BI (Blocking Island) paradigm [11]. BI provides an efficient way of abstracting resource (e.g., bandwidth) available in a network. BI is a cluster constructed according to the bandwidth availability. β-BI means a cluster in which links composing intra–cluster routes for node–pairs inside have β or more bandwidth.

Our algorithm constructs β-BIs by repeating merge operation. There are two differences between the original BI and ours. First, the size of a BI (i.e., a cluster) is bounded in our clustering problem. To maximize the bandwidth from an ingress to an egress border node in a BI, each BI should consist of links with more bandwidth. We realize this by making each cluster give higher priority in taking links with more bandwidth in. Second, we need to bound the maximum node–hop count from an ingress to an egress border node in each BI. This is to prevent the blocking probability from increasing because of increased node–hop count of a lightpath.

The following lists symbols we use in our proposed algorithm.

- B : Upper bound for number of nodes that each cluster includes.
- β : Lower bound for the number of fibers on links that are taken in clusters.
- H : Upper bound for the node–hop counts from an ingress to an egress border nodes in each cluster.
- T_w : Waiting time for merge requests to arrive. Each cluster does a merge operation that is requested within T_w.
- R_s : Minimum number of lightpaths available between border nodes in cluster V_s.
- R_{st} : Minimum number of lightpaths available on links between cluster V_s and V_t.
- $V_{s\cup t}$: Cluster into which cluster V_s merges cluster V_t.

Now, we will present our algorithm, where each cluster V_i individually performs a merge operation. When a hierarchy is not introduced (i.e., no cluster is constructed), each node is regarded as a cluster. When a node is added to the network, the node is regarded as a cluster.

- Step 1: Border nodes in V_i set T_w and wait for merge requests from adjacent clusters. Go to Step 2 in time T_w.
- Step 2: The border nodes in V_i exchange a received merge request among them. If one or more merge requests arrive, then go to Step 3. Otherwise, go to Step 5.
- Step 3: The border nodes in V_i select V_t that sent a merge request with the maximum effect among clusters that sent a merge request to V_i. If there exist more than one candidates for V_t, the cluster having the greatest cluster

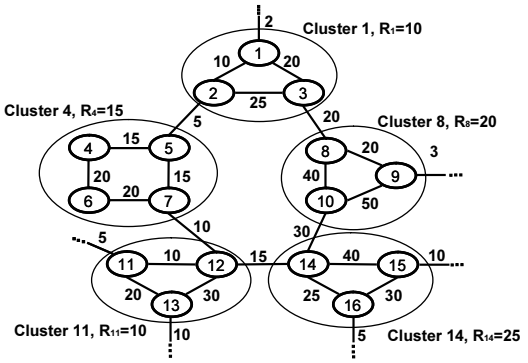


Fig. 5 Before merge operation.

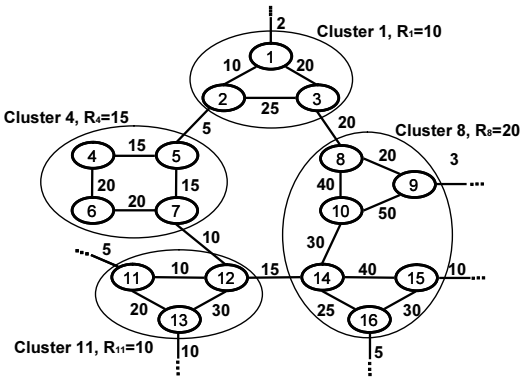


Fig. 6 V_{14} merges with V_8.

ID is selected as V_t [†]. The effect of a merge operation is calculated as $\min(R_i, R_{it}, R_t)$, which is included in a request message. P_i, which is the border node that received the merge request from V_t, sends an *accept merge request* message to V_t. Border nodes that received a merge request from adjacent clusters except V_t send a *refuse merge request* message to the senders of merge requests. Go to Step. 4.

Step 4: P_i informs all nodes in V_i of accepting a merge request. All nodes update (1) node–cluster matching information (change the cluster ID of nodes in $\max(V_i, V_t)$ to $\min(V_i, V_t)$), (2) intra–cluster route information, (3) border node information (whether each node is a border node or not), and (4) $R_{i\cup t}$. Then, border nodes advertise new node–cluster matching information to other clusters. Go back to Step 1.

Step 5: Among adjacent clusters, select $V_{t'}$ such that $\min(R_i, R_{it'}, R_{t'})$ is maximized while satisfying (1) the size of $V_{i\cup t'}$ is B or less, (2) $\min(R_i, R_{it'}, R_{t'}) \geq \beta$, and (3) the maximum node–hop count of intra–route from an ingress to egress node in $V_{i\cup t'}$ is H or less. If there exist more than one candidates for $V_{t'}$, the cluster having the greatest cluster ID is selected as $V_{t'}$. The above selection is done by exchanging information among border nodes in V_i. A border node that is adjacent to $V_{t'}$ and whose node ID is maximum is selected as P_i, which requests a merge operation. If there exists P_i, P_i sends a *merge request* message to $V_{t'}$ and go to Step 6. Otherwise, go to Step 7.

Step 6: If P_i receives an *accept merge request* from $V_{t'}$, P_i informs all nodes in V_i of succeeding in merge request. All nodes update (1) node–cluster matching information (change the cluster ID of nodes in $\max(V_i, V_{t'})$ to $\min(V_i, V_{t'})$), (2) intra–cluster route information, (3) border node information (whether each node is a border node or not), and (4) $R_{i\cup t'}$. Then, border nodes advertise new node–cluster matching information to other clusters. Go back to Step 1. Otherwise (P_i receives a *refuse merge request*), P_i informs all nodes in V_i of failing in merge request and go to Step 1.

Step 7: Border nodes in V_i advertise new inter–cluster route information. Then, finish this algorithm because there are no adjacent clusters that V_i can perform merge operation with.

In Step 3 and 5, when there exist more than one candidates for V_t ($V_{t'}$), we use the greatest cluster ID as a tie–break condition. This is because border nodes in the same cluster can uniquely determine V_t ($V_{t'}$). If other selection policies (e.g., random) are adopted, border nodes need to exchange additional information to negotiate which cluster each border node selects.

In trying to perform a merge operation, border nodes in V_i approximately calculate $R_{i\cup t}$ as $\min(R_i, R_{it}, R_t)$. Let us now explain why $R_{i\cup t}$ is $\min(R_i, R_{it}, R_t)$. The border node pair where the number of available lightpaths is minimum belongs to (1) V_i, (2) V_t, or (3) both V_i and V_t. In (1) and (2), the minimum number of lightpaths corresponds to R_i and R_t, respectively. In (3), the route between a border node in V_i and one in V_t consists of the route between border nodes in V_i, the link between V_i and V_t, and the route between border nodes in V_t. Thus, the minimum number of lightpaths on these routes and the link, that is, $\min(R_i, R_{it}, R_t)$, corresponds to $R_{i\cup t}$. Note that $R_{i\cup t}$ does not always equal $\min(R_i, R_{it}, R_t)$ because all links between V_i and V_t are not always part of the routes between border nodes in cluster $V_{i\cup t}$. However, V_i do not calculate $R_{i\cup t}$ precisely because this calculation needs

[†]The smallest cluster ID is an alternative tie–break condition. We examined it by computer simulation, but the resulted performance was almost the same.

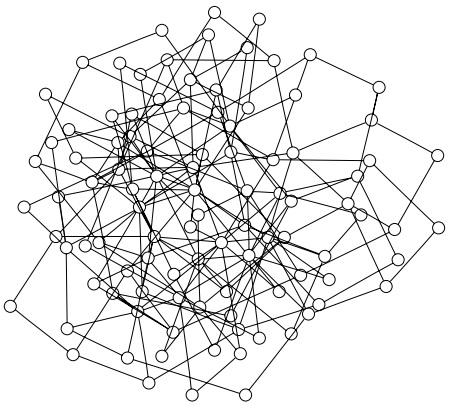


Fig. 7 Random network ($N = 100$).

hop counts for all the routes between all the border node pairs, which degrades the scalability of our clustering method.

Figures 5 and 6 have samples of a merge operation. We set the number of wavelengths multiplexed on fibers to one for the sake of simplicity. When cluster 14 merges with cluster 11 in Fig. 5, the minimum number of lightpaths available between border nodes, $R_{14\cup11}$ is equal to $\min(R_{14}, R_{14,11}, R_{11}) = \min(25, 15, 10) = 10$. When cluster 14 merges with cluster 8, $R_{14\cup8} = 20$. Since $R_{14\cup8} > R_{14\cup11}$, cluster 14 sends a merge request to cluster 8. Figure 6 depicts the construction of clusters after cluster 14 merges with cluster 8. The route from cluster 11 to cluster 1 changes from $12 \rightarrow 7 \rightarrow 5 \rightarrow 2$ to $12 \rightarrow 14 \rightarrow 10 \rightarrow 8 \rightarrow 3$. If there are some candidate routes with the same cluster–hop counts, we select a route where the number of available lightpaths is maximum. Note that the number of lightpaths available on the route changes from 5 to 15.

4. Evaluation Results

4.1 Simulation Model

We used random networks with 100, 200, 300, 400, and 500 nodes generated by the Waxman algorithm [12] whose parameters α and β were 0.15 and 0.2, respectively. Fig. 7 shows the resulting random network with 100 nodes. We assume that there is no propagation delay on each link and no processing delay on each node. Note, however, that even if propagation delay and processing delay are considered, the resulted clustering is identical as long as the time for a pair of clusters to complete a merge operation is smaller than T_w. The number of fibers on link uniformly ranged from 1 to 30. There were 32 wavelengths multiplexed on a fiber.

We compared our distributed clustering method applied to the *bounded, connected, max–lightpath* problem (BI) with (1) a network without any clusters (no cluster) and (2) a distributed clustering method applied to the *bounded, connected, min–cut* problem (min–cut). With min–cut, we set the link cost to 1. In this case, each cluster merges an adjacent cluster that has the maximum number of connected links, which leads to maximizing the number of links in merged clusters (i.e., minimizing the number of links between clusters).

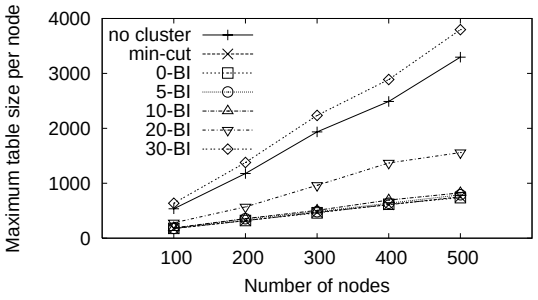


Fig. 8 Maximum table size maintained by node.

Table 1 Average number of clusters constructed.

no cluster	min-cut	0-BI	5-BI	10-BI	20-BI	30-BI
100	11.5	12.6	14.5	19.4	43.5	100

With each clustering method, we set B to $\sqrt{N}$ (N was the number of nodes in the network) because setting B to $\sqrt[M]{N}$ in a network with M layers leads to minimized table length [9] and $M = 2$. We set H to $\sqrt{N}$ (upper bound on H) because small H may not lead to increasing the number of lightpaths available between nodes. The waiting time for a merge request, T_w, was set to $\gamma \times T$. γ was a uniform random variable from 1 to 4 and $T = 10(s)$, which was large enough for a merged cluster to update each piece of information in the cluster.

4.2 Maximum Table Size Maintained by Node

Figure 8 shows the maximum table size maintained by a node in the networks with different numbers of nodes. In networks without clusters, each node only maintains a routing table that has a set of routes to all nodes. In clustered networks constructed with BI and min–cut, on the other hand, each node maintains a node–cluster mapping table and an intra–cluster routing table (see Sec. 3.2). In addition, each border node maintains an inter–cluster routing table. We defined the table size as the total hop count of routes for intra/inter–cluster routing tables and as the total number of entries for a node–cluster mapping table. In our BI, β is set to 0, 5, 10, 20, and 30. 30-BI does not perform merge operation because there exists no link that has more than 30 fibers.

0-BI and min-cut show the smaller table size than others because merge operation is not limited by the constraint about β in those methods. The table sizes in 0-BI and min–cut are about between 22% and 33% of that without clusters. This is because 0-BI and min–cut reduce the number of routes by aggregating routes to nodes in the same cluster. As the number of nodes increases, the effect of aggregation increases.

0-BI yields almost the same table size as min–

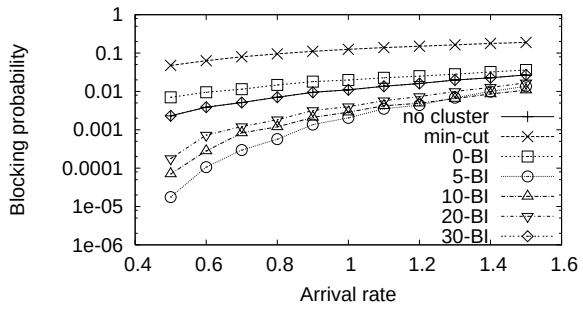


Fig. 9 Blocking probability for lightpath requests (holding time: 60s).

cut does because the numbers of clusters and nodes included by each cluster with both methods are similar. 30-BI needs more memory than that without clusters. This is because 30-BI has node-cluster mapping table in addition to inter-cluster routing table that is same as the routing table in the network without clusters.

As for BI, the table size increases as β gets larger. This is because larger β limits the number of merge operations performed in the network. As a result, less routes are aggregated. Table 1 shows the average number of clusters constructed with each method in the network with 100 nodes. As more merge operations are performed, the number of clusters constructed gets close to the optimal value ($\sqrt{N} = 10$). When β is relatively small ($\beta = 5$), the table size can be reduced close to the minimum size since most merge operation are not limited by constraint as to β.

4.3 Blocking Probability for Lightpath Requests

We next evaluate the blocking probability for lightpath requests. Lightpath requests arrive after the clusters are constructed. The requests arrive according to a Poisson process at a rate of λ (requests/s) and the holding time for lightpaths follows an exponential distribution with an average of 60 seconds. From here, we use a random network with 100 nodes. The results are shown in Fig. 9. The horizontal axis represents the arrival rate of lightpath requests and the vertical axis represents the blocking probability for lightpath requests.

In Fig. 9, the results by BIs outperform the results by min–cut for all arrival rates. This is because more wavelength resources are provided for each node–pair in BIs. Comparing BIs with different β, 5-BI shows the lowest blocking probability among them. Before we explain why 5-BI shows good performance, we show the average number of lightpaths available between nodes in Tab. 2, the maximum load on link in Tab. 3, and the average number of node–hop counts of lightpaths in Tab. 4. Here, we define the load on channel as the ratio of the number of node–pairs that traverses the link to the number of wavelengths on the link. From these tables, we observe that more lightpaths available between nodes make the blocking probability lower while the average hop–count increases by constructing clusters. However, this is not enough. Requests of lightpaths through heavy–load link tends to be rejected, which makes the overall blocking probability increases. Therefore, minimizing the maximum load is also important for decreasing the blocking probability.

Now we explain why 5-BI shows the lowest blocking probability among other algorithms. The reject of lightpath request tends to occur on links with few fibers. To decrease the blocking probability, the number of node–pairs that traverse those links must be minimized. 5-BI realizes this by 1) locating links less than five fibers between clusters, and 2) constructing clusters whose sizes are near to B. As the size of cluster gets larger, the cluster tends to have more links between adjacent clusters. If there are several links between clusters, the link with more fibers is selected as an inter–cluster route. The other links with few fiber are not selected as an inter–cluster route. In 30-BI, the size of each cluster is one and each cluster has only one link between an adjacent cluster. The sizes of clusters in 10-BI and 20-BI are smaller than that in 5-BI. As a result, 10-BI and 20-BI show higher blocking probability than 5-BI does. In 0-BI, each cluster can include links with few fibers, which leads to higher blocking probability.

We conclude that 5-BI provides better performance in terms of blocking probability than others while keeping the routing table size almost the same as 0-BI and min-cut.

4.4 Adaptation to the Topology Change

We further evaluate our 5-BI-based clustering method when a new node is added to network. In this case, the reconstruction of clusters is needed. To realize this, we introduce a *give* operation, in which a cluster gives one of its border nodes to an adjacent cluster. A give operation is performed when a cluster cannot perform a merge operation. Cluster V_i gives its border node to adjacent cluster V_t if all the following six conditions are satisfied: (1) the size of V_t is $B-1$ or less, (2) the size of V_i is more than 1, (3) the maximum node–hop count of intra–route from an ingress to egress node in V_t is $H-1$ or less, (4) R_i increases, (5) R_{it} decreases, and (6) V_i remains connected. V_i selects a cluster (say V_t) among adjacent clusters such that the increase in R_i is maximized. It is better to increase both R_i and R_t. However, V_i cannot know the increase in R_t before the give operation because detail of intra–cluster route information of R_j is not available.

We compare three kinds of clustering methods: (1) BI-*scratch* where new clusters are constructed from scratch when a new node is added, (2)BI-*incremental merge* where the existing clusters and a new cluster (a new node) try to perform only the merge operation, and (3) BI-*give* where the existing clusters and a new cluster try to perform both merge and give operations.

Table 2 Average number of lightpaths available between nodes.

no cluster	min-cut	0-BI	5-BI	10-BI	20-BI	30-BI
309.9	243.2	334.5	353.2	358.2	353.6	309.9

Table 3 Maximum load on channel.

no cluster	min-cut	0-BI	5-BI	10-BI	20-BI	30-BI
2.55	7.41	4.14	1.70	1.91	2.27	2.55

Table 4 Average number of node–hop counts of lightpaths.

no cluster	min-cut	0-BI	$5-BI$	10-BI	20-BI	30-BI
3.33	4.74	4.63	4.60	4.51	4.18	3.33

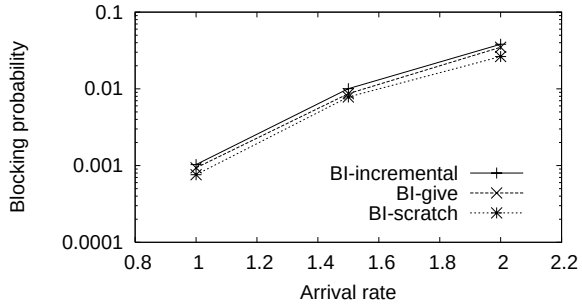


Fig. 10 Blocking probability for lightpath requests (21 nodes are added).

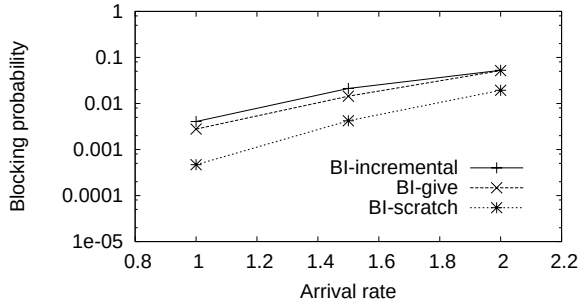


Fig. 11 Blocking probability for lightpath requests (44 nodes are added).

Figure 10 shows the blocking probability when 21 nodes are added one by one to a network with 100 nodes. BI-incremental and BI-give achieve almost the same blocking probability as BI-scratch in spite that BI-incremental and BI-give performs much smaller number of operations than BI-scratch.

Figure 11 shows the blocking probability when 44 nodes are added one by one. When more nodes are added, BI-give shows lower blocking probability than BI-incremental. This is because give operation increases the number of wavelengths available in clusters and releases links with few fibers out of cluster.

However, BI-give does not achieve as low blocking probability as BI-scratch does when 44 nodes are added. This means that the number of added nodes that give operation can cope with is limited. When give operation is not effective, we need to reconstruct clusters from scratch. To determine when we should perform the reconstruction instead of give operation is important, but it is left for our future work.

5. Conclusions

We proposed a distributed node–clustering method for hierarchical routing in lightpath networks. The method based on Blocking Island paradigm maximizes the number of lightpaths between nodes. Throughout our simulation, we found that the table size with our BI with appropriate β ranged between 22% and 33% of that in a cluster–less network. The effect of aggregating the route information increased as the number of nodes increased. In terms of the blocking probability for lightpath requests in a network with 100 nodes, we found that locating links with fewer fibers between clusters was important in addition to increasing the number of lightpath in cluster for decreasing blocking probability. We further evaluated a method to restructure clusters (give operation) when new nodes are added to a network. We found that our give operation is effective until a certain number of nodes are added.

There is a tradeoff between the number of performed clustering operations and the performance of clusters when topology changes. We determine when to perform the reconstruction instead of give operation in future work.

Acknowledgments

This work was supported in part by "The 21st Century Center of Excellence Program", and by a Grant-in-Aid for Scientific Research (A) 14208027 from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

[1] K. Kompella and Y. Rekhter, "OSPF extensions in support of generalized MPLS," *Internet Draft* draft-ietf-ccamp-ospf-gmpls-extensions-12.txt, Oct. 2003.
[2] Y. Rekhter and T. Li, "A Border Gateway Protocol 4 (BGP-4)," *IETF RFC 1771*, Mar. 1995.
[3] L. Berger, "Generalized multi-protocol label switching (GMPLS) signaling resource reservation protocol-traffic engineering (RSVP-TE) extensions," *IETF RFC 3473*, Jan. 2003.
[4] G. Bernstein, B. Rajagopalan, D. Pendarakis, A. Chiu, J. Strand, V. Sharma, D. Cheng, R. Izmailov, L. Ong, and

S. Dharanikota, "Optical inter domain routing considerations," *Internet Draft* draft-ietf-ipo-optical-inter-domain-02.txt, Feb. 2003.

[5] D. Wang, J. Strand, J. Yates, C. Kalmanek, G. Li, and A. Greenberg, "OSPF for routing information exchange across metro/core optical networks," *Optical Networks Magazine*, vol. 3, pp. 34–43, Sept. 2002.

[6] M. J. Francisco, S. Simpson, L. Pezoulas, C. Huang, J. Lambadaris, and B. St. Arnaud, "Interdomain routing in optical networks," in *Proceedings of Opticomm2001*, pp. 120–129, Aug. 2001.

[7] C. Pelsser and O. Bonaventure, "Extending RSVP-TE to support inter-AS LSPs," in *Proceedings of 2003 Workshop on High Performance Switching and Routing (HPSR 2003)*, pp. 79–84, June 2003.

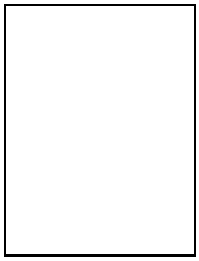
[8] W. Li, "Inter-domain routing: Problems and solutions," *Technical Report TR-128, Department of Computer Science, State University of New York*, Feb. 2003.

[9] L. Kleinrock and F. Kamoun, "Hierarchical routing for large networks," *Computer Networks*, vol. 1, pp. 155–174, Jan. 1977.

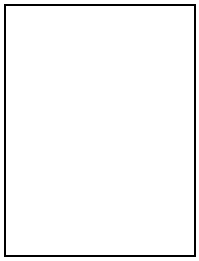
[10] R. Krishnan, R. Ramanathan, and M. Steenstrup, "Optimization algorithms for large self-structuring networks," in *Proceedings of IEEE INFOCOM'99*, pp. 71–78, Mar. 1999.

[11] D. Zhemin and M. Hamdi, "Resource management in multi-segment optical networks using the blocking island paradigm," in *Proceedings of 2003 Workshop on High Performance Switching and Routing (HPSR 2003)*, pp. 43–48, June 2003.

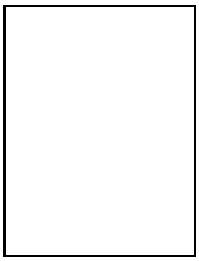
[12] B. M. Waxman, "Routing of multipoint connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, pp. 1617–1622, Dec. 1988.



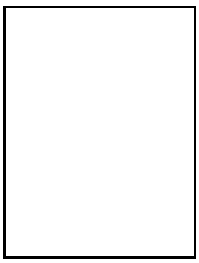
Yukinobu FUKUSHIMA received the B.E. and M.E. degrees from Osaka University, Japan, in 2001 and 2003, respectively. He is currently a Ph.D. student of Graduate School of Information Science and Technology, Osaka University. His research interest includes network design and routing in photonic networks. He is a member of IEICE.



Hiroaki HARAI received the M.E. and Ph.D. degrees in information and computer sciences from Osaka University, Osaka, Japan, in 1995 and 1998, respectively. In April 1998, he joined Communications Research Laboratory (currently NICT), Tokyo, Japan. He is currently a Senior Researcher of National Institute of Information and Communications Technology (NICT), Tokyo, Japan. His current research topic is to develop optical networks, especially lightpath networks for computer communications. His research interest also includes optical packet switching and routing. He is a member of IEEE and IEICE.



Shin'ichi ARAKAWA received the M.E. and D.E. degrees in Informatics and Mathematical Science from Osaka University, Osaka, Japan, in 2000 and 2003, respectively. He is currently a Research Assistant at the Graduate School of Economics, Osaka University, Japan. His research work is in the area of photonic networks. He is a member of IEEE and IEICE.



Masayuki MURATA received the M.E. and D.E. degrees in Information and Computer Sciences 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 Osaka University, and is now with Graduate School of Information Science and Technology, Osaka University since April 2004. He has more than three hundred papers of international and domestic journals and conferences. His research interests include computer communication networks, performance modeling and evaluation. He is a member of IEEE, ACM, The Internet Society, IEICE and IPSJ.