Distributed Clustering Method for Large-Scaled Wavelength Routed Networks

Yukinobu Fukushima†, Hiroaki Harai‡, Shin'ichi Arakawa£, and Masayuki Murata†
†Graduate School of Information Science and Technology, Osaka University, 1-5 Yamadaoka, Suita, Osaka 560-0871, Japan
‡National Institute of Information and Communication Technology, 4-2-1 Koganei, Tokyo 184-8795, Japan
£Graduate School of Economics, Osaka University, 1-7 Machikaneyama, Toyonaka, Osaka 560-0043, Japan
Email: {y-fukusm, murata}@ist.osaka-u.ac.jp, harai@nict.go.jp, arakawa@econ.osaka-u.ac.jp

***Abstract*— 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.**

keywords– **WDM, lightpath network, path–vector routing, hierarchical routing, distributed clustering**

I. Introduction

WDM lightpath networks are one of the most promising candidates for the next generation Internet. When traffic demand occurs, a lightpath, where signals are handled optically at intermediate nodes, is configured to transport this 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, 2] or BGP [3]. Then, reservation protocols such as RSVP–TE [4] 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 [5-8]. Bernstein *et. al* [5] 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 [7]) as the inter–domain routing and wavelength reservation protocol.

Wei *et. al* [9] 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* [10]. The basic idea behind hierarchical routing is to form a set of nodes into a *cluster* to aggregate route information about nodes far from the 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 routing table size, 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.

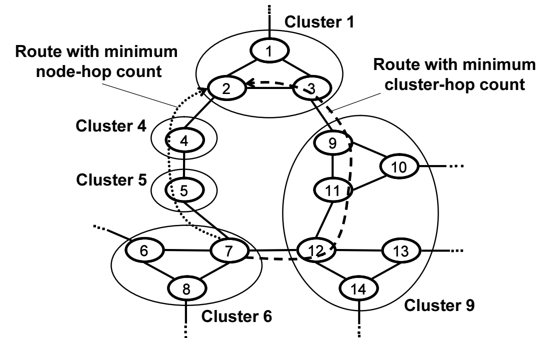

Fig. 1. Route with minimum cluster–hop count and route with minimum node–hop count.

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 maximized the number of lightpaths between nodes. The key idea behind our method is to construct the node–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 II discusses hierarchical routing, node clustering and the conventional clustering problem. In Sec. III, we propose a distributed method of clustering for WDM lightpath networks. Section IV presents evaluation results obtained by simulation. Finally, we present our conclusions and the directions of future work in Sec. V.

II. Hierarchical Routing and Node Clustering

A. 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

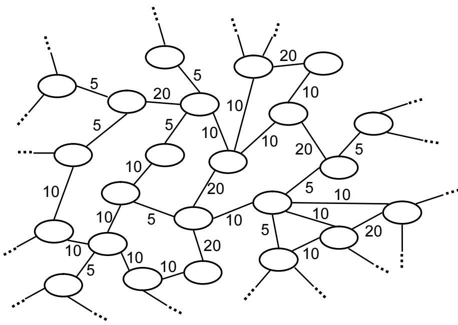

Fig. 2. Network Model.

BGP for the routing protocol since it meets the requirements of the inter–domain routing protocol in the optical networks [5]. In each intermediate domain, an intra–domain lightpath is set up according to a routing/signaling protocol adopted in the domain.

B. Hierarchical Clustering

Figure 3 has an example of hierarchical clustering. We call a set of nodes a cluster. A node whose adjacent node belongs to another cluster is referred to as a *border node*. An 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.

C. Conventional Clustering Problem

Krishnan *et. al* [11] 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.

Bounded, connected, min–cut problem

Given,

- A undirected graph $G = (V, E)$ with edge weights $w : E \rightarrow Z_0^+$, and
- 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

$$minimize \sum_{e} w(e) \qquad (1)$$

where $e \in E$, $e \notin E_i$, $i \in \{1, 2, \ldots, k\}$ $\forall k \in \{2, \ldots, |V|\}$.
Following restrictions need to be satisfied,

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

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

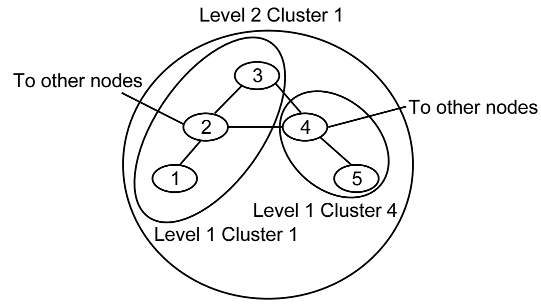

Fig. 3. Example hierarchical clustering.

This is because the construction of balanced clusters does not always result in minimized link costs between clusters.

The computational complexity of the *bounded, connected, min–cut* problem is NP–complete. Therefore, Rajesh *et. al* [11] proposed a centralized heuristic algorithm to solve this problem, which 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.

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

A. Node Clustering in WDM Lightpath Networks

As we discussed in Section I, 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 now explain how these requirements are satisfied with our distributed algorithm.

We believe that increasing the number of lightpaths available between nodes in WDM lightpath networks will lead to decreased blocking probability in lightpaths. Based on this idea, 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 $F_{ii} = 0, (\forall i = 1, \ldots, N)$.

Constrains:

- 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. Note that we do not consider wavelength conversion in this paper. If wavelength conversion is allowed, we need to weight the number of lightpaths available on a route with wavelength converters because we can set up more lightpaths on the route than that on routes without wavelength converters.

Our *bounded, connected, max–lightpath* problem is NP–complete since our problem determines whether a given graph is partitioned into bounded and connected partitions, which is called as *bounded, connected, partitioning* problem and shown to be NP–complete [11]. We therefore 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", which prevents intra/inter–cluster routing tables from becoming too large. 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. IV, 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. III-B, we will explain our algorithm in Sec. III-C.

B. 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*. The intra–cluster routing table is constructed by route information exchanged among nodes that belong to the same cluster, and inter–cluster routing table is constructed by route information exchanged among all the border nodes. 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 (a) to obtain its cluster identifier, and (b) 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

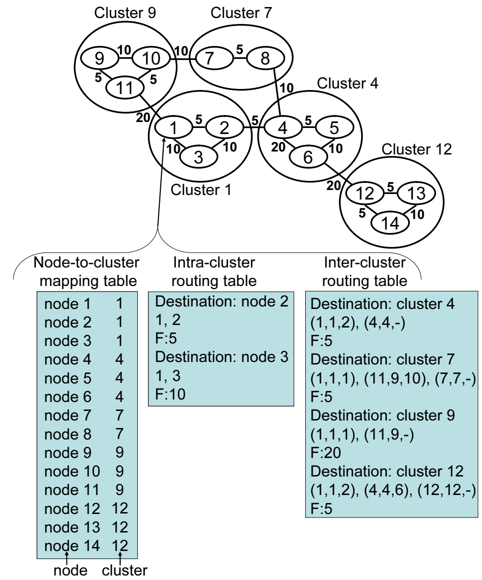

Fig. 4. (1) Node–to–cluster mapping table, (2) inter-cluster routing table, and (3) intra-cluster routing table.

   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) 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 (b) the number of fibers available on the 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.

C. Distributed Clustering Algorithm for Bounded, Connected, Max–lightpath Problem

Our algorithm constructs clusters by repeating a *merge* operation, which makes a cluster merge with one that is adjacent. Each cluster selects 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. To achieve this, we tried to maximize

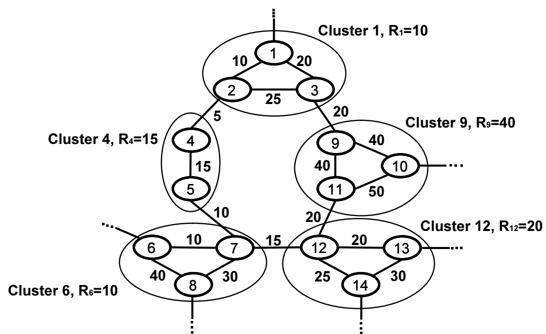

Fig. 5. Before merge operation.

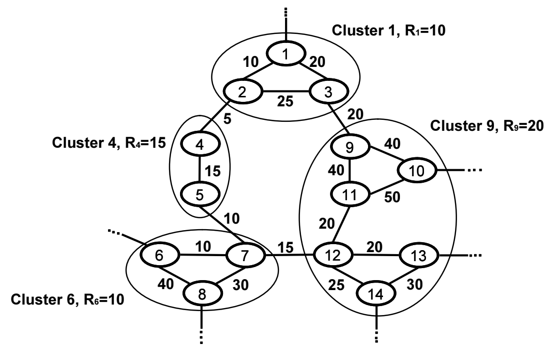

Fig. 6. After merge operation.

the number of lightpaths available from an ingress to an egress border node in each cluster, which leads to the number of lightpaths between nodes in the different clusters being maximized.

The following lists symbols we use in our proposed algorithm.

- B : Upper bound for number of nodes that each cluster includes.
- 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.
- T_w : Waiting time for merge requests to arrive. Each cluster does a merge operation that it receives within T_w.
- $V_{s\cup t}$: Cluster into which cluster V_s merges cluster V_t.

Now, we will present our algorithm, where each cluster V_i individually does 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. 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 and new inter–cluster route 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 that the size of $V_{i\cup t'}$ is B or less. The above selection is done by exchanging information among border nodes in V_i. A border node that requests a merge operation is selected as P_i. 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 and new inter–cluster route 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: Finish this algorithm because there are no adjacent clusters that V_i can perform merge operation with.

We 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, we do not calculate $R_{i\cup t}$ precisely because this calculation needs 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 merge operations. We set the number of wavelengths multiplexed on fibers to one for the sake of simplicity. When cluster 12 merges with cluster 6 in Fig. 5, the minimum number of lightpaths available between border nodes, $R_{12\cup 6}$ is equal to $\min(R_{12}, R_{12,6}, R_6) = \min(20, 15, 10) = 10$. When cluster 12 merges with cluster 9, $R_{12\cup 9} = 20$. Since $R_{12\cup 9} > R_{12\cup 6}$, cluster 12 sends a merge request to cluster 9. Figure 6 depicts the construction of clusters after cluster 12 merges with cluster 9. The route from cluster 6 to cluster 1 changes from $7 \to 5 \to 4 \to 2$ to $7 \to 12 \to 11 \to 9 \to 3$. 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.

IV. Evaluation results

A. 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. We set the propagation delay for each link to 10ms. 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 (max–lightpath) 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 tried to minimize the number of links between clusters (i.e, set the link cost at a constant value).

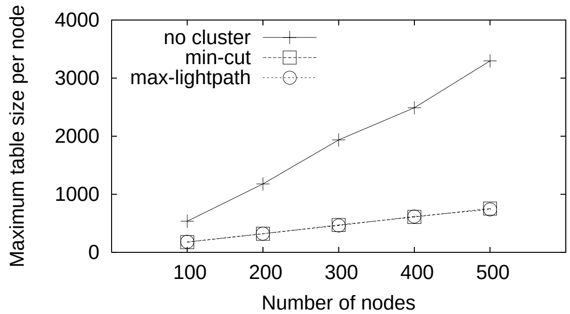

Fig. 7. Maximum table size maintained by node.

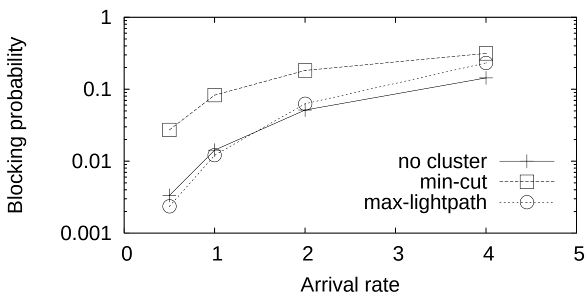

Fig. 8. Blocking probability for lightpath requests (holding time: 60s).

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 [10] and $M = 2$. 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.

B. Maximum Table Size Maintained by Node

Figure 7 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 max–lightpath and min–cut, on the other hand, each node maintains a node–cluster mapping table and an intra–cluster routing table (see Sec. III-B). 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 Fig. 7, the table sizes in max–lightpath and min–cut are between 22% and 33% of that without clusters. This is because max–lightpath 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. Max–lightpath yields almost the same table size as min–cut does because the numbers of clusters and nodes included by each cluster with both methods are similar.

C. 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 60s.

In Fig. 8, the horizontal axis represents the arrival rate of lightpath requests and the vertical axis represents the blocking probability for lightpath requests. We use a random network with 100 nodes. When load is low (arrival rate of less than 1), the blocking probability for max–lightpath is almost the same as that without clusters. This is because max–lightpath increases the number of lightpaths available between nodes. Table I lists the average number of lightpaths available between nodes with each method. From Fig. 8 and Tab. I, we can see that increasing the number of lightpaths available between nodes leads to decreased blocking probability for lightpath requests.

TABLE I
AVERAGE NUMBER OF LIGHTPATHS AVAILABLE BETWEEN NODES.

no cluster	min-cut	max-lightpath
307.8	262.6	337

As the load increases, network without clusters outperforms max–lightpath. This is because the node–hop count of lightpaths with max–lightpath is larger than that without clusters, which results in each lightpath with max–lightpath wasting more resources than network without clusters. As a result, more links are over–loaded with max–lightpath than those without clusters. The average node–hops counts of lightpaths are 3.33, 4.6, and 4.47 for without clusters, max–lightpath, and min–cut, respectively. However, max–lightpath outperforms min–cut for all loads by providing each node–pair with more available lightpaths.

V. CONCLUSIONS

We proposed a distributed node–clustering method (max–lightpath) that achieves hierarchical routing in lightpath networks. Throughout our simulation, we found that the table size with our max–lightpath 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 (1) our max–lightpath attains lower blocking probability than min–cut, which is a distributed clustering algorithm for the conventional problem and (2) reduces the blocking probability as low as that in a network without clusters when loads are low.

We only treated the routing table size as a scalability problem. However, the volume of messages entailed in exchanging information poses another scalability problem. We intend to evaluate the volume of messages in future work. In addition, we intend to propose a new clustering operation to restructure the network when the network topology changes (e.g., new nodes join the network).

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] C. Huitema, *Routing in the Internet (2nd Edition)*. Prentice Hall, 2000.
[3] Y. Rekhter and T. Li, “A Border Gateway Protocol 4 (BGP-4),” *IETF RFC 1771*, Mar. 1995.
[4] L. Berger, “Generalized multi-protocol label switching (GMPLS) signaling resource reservation protocol-traffic engineering (RSVP-TE) extensions,” *IETF RFC 3473*, Jan. 2003.
[5] 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.
[6] 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.
[7] 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.
[8] 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.
[9] W. Li, “Inter-domain routing: Problems and solutions,” *Technical Report TR-128, Department of Computer Science, State University of New York*, Feb. 2003.
[10] L. Kleinrock and F. Kamoun, “Hierarchical routing for large networks,” *Computer Networks*, vol. 1, pp. 155–174, Jan. 1977.
[11] 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.
[12] B. M. Waxman, “Routing of multipoint connections,” *IEEE Journal on Selected Areas in Communications*, vol. 6, pp. 1617–1622, Dec. 1988.