Dynamic Multicast Traffic Grooming in Optical WDM Mesh Networks: Lightpath vs Light-tree

Xiaojun Yu, Gaoxi Xiao, and Tee-Hiang Cheng

Abstract— With the rising popularity of multicast applications, various algorithms using either lightpath or light-tree scheme have been proposed for dynamic multicast traffic grooming in meshed wavelength division multiplexing (WDM) networks. To the best of our knowledge, however, no systematic comparison has ever been made between the performances of the two schemes in minimizing network blocking probability. In this paper, we address the dynamic multicast traffic grooming problem in WDM networks, and present comprehensive comparisons between these two schemes in different cases. Our main contributions are two-fold: first, we compare the performances of the existing lightpath and light-tree based grooming algorithms, and show that in most cases, the lightpath-based methods outperform the light-tree based ones. We discuss and explain such observations. Second, we propose a new lightpath-based algorithm, named LightPath Fragmentation (LPF) method, to further improve the network blocking performance. Numerous simulations show that the LPF method steadily outperforms the existing algorithms in different cases. Effects of the ratio of unicast traffic loads versus overall traffic loads and the average number of destinations of each multicast request are also studied.

Index Terms — Optical networks, multicast, dynamic traffic grooming, lightpath, light-tree

I. INTRODUCTION

With the explosive increase in Internet traffic loads over the past few decades, WDM networks have emerged to be the dominant infrastructure for backbone networks [1]. In wavelength-routed WDM networks, all-optical communication channels, referred to as *lightpaths,* can be established with the help of optical cross-connects (OXCs) [2]. A lightpath may span several physical links, and if no wavelength converter is utilized, it has to be provisioned with the same wavelength along its route, which is known as *wavelength continuity constraint* [3].

Multicast is an efficient way to transmit information from one source to multiple destinations simultaneously. As more and more multicast applications such as multiparty conferencing, video distribution and HDTV etc. are becoming increasingly popular, multicast traffic is expected to constitute a large portion of the overall network traffic in the future. To support multicasting on the physical layer of WDM networks, a new concept named *light-tree* has been introduced [4]. A light-tree is an all-optical channel that supports data transmission from a single source to multiple destinations, typically utilizing the optical splitting capability of its branching nodes.

The bandwidth needed for a multicast session typically ranges from several to tens of megabits per second (Mbps), which is much lower compared to the 2.5-40 gigabits per second (Gbps) capacity that can be steadily provided by a single wavelength channel in today's WDM networks. To efficiently utilize the wavelength capacity, *traffic grooming* is usually adopted [5]. From an algorithm point of view, there are static and dynamic traffic grooming problems. For the static problems, the traffic demands are known in advance and the main objective is generally to minimize the cost for supporting all the demands, or to support as many demands as possible with the given resources. For the dynamic problems, the unicast/multicast demands arrive and leave dynamically. The main objective of the algorithm design is therefore typically to minimize the blocking probability or to maximize the network throughput.

The early-stage work on multicast traffic grooming was mainly for tackling the static problems [6-9]. With the developments of optical communication technologies, dynamic multicast traffic grooming problems become increasingly important and quite a few algorithms have been proposed [10-19]. Such algorithms typically utilize either lightpaths or light-trees, or both of them, to support dynamic multicast traffic. For convenience, we term those algorithms utilizing only the lightpaths for multicast traffic grooming as *lightpath based* approaches; and *light-tree based* approaches otherwise. Though some rather simple comparisons between these two groups of different approaches have been made for multicast transmission [10] and many-to-many transmission [11], to the best of our knowledge, there is no systematic comparison between their respective blocking performances.

In this paper, we address the dynamic multicast traffic grooming problem in WDM networks with the main objective of minimizing the network *bandwidth blocking ratio* (BBR), which is defined as

Manuscript received December 15, 2012; revised March 14, 2013.

The authors are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore (phone: +65-67904552; fax: +65-67933318; e-mail: egxxiao@ ntu.edu.sg).

$$BBR = \frac{\sum \text{bandwidth of blocked requests}}{\sum \text{bandwidth of all requests}}.$$

We first compare the blocking performances of various existing lightpath and light-tree based approaches and show that the lightpath based approaches generally have a winning margin. Then we propose a new lightpath-based method called *LightPath Fragmentation* (LPF) method. Extensive simulations results show that LPF outperforms all the existing algorithms in different cases. Effects of other factors, including the average number of destinations of each multicast session and the ratio of unicast traffic loads to overall traffic loads etc., are also studied.

The remainder of this paper is organized as follows. Section II briefly reviews the existing lightpath and light-tree based algorithms for dynamic multicast traffic grooming. Section III presents a brief comparison between the existing approaches and explains why the lightpath based approaches have a winning margin. Section IV describes the proposed LPF algorithm. Simulation results are presented in Section V. Section VI concludes the paper.

II. Previous Work

A. Lightpath Based Dynamic Mulitcast Traffic Grooming

Authors in [12] proposed the first lightpath based dynamic multicast traffic grooming algorithm, named Maximizing Minimum Freeload (MMFL) method. To simplify the calculations, MMFL limits the route selection for each multicast request to be on a single wavelength which, among all the wavelengths, has the largest overall residual capacity after accommodating the multicast request. The algorithm is simple, but the single-wavelength constraint degrades the network resource utilizations.

To alleviate the unfavorable single-wavelength constraint, two other algorithms, which we term as logical-path-tree (LPT) method and saturated cut (SC) method respectively, were proposed in [13]. Both algorithms adopt the same main idea of utilizing existing logical links to serve as many destinations as possible before setting up new lightpaths. Between them, SC achieves better performance by first finding islands which include at least one of the sources/destinations and other nodes connected to them via existing links with sufficient residual capacities, and then connecting such islands by setting up new lightpaths. To the best of our knowledge, SC performs the best among the existing lightpath based methods. Thus it will be adopted for comparisons in this paper.

B. Light-tree Based Dynamic Multicast Traffic Grooming

Depending on whether an established ligth-tree can be changed or not in grooming, algorithms of this category can be classified into *fixed* light-tree methods and *adaptive* ones.

For the fixed light-tree methods, an established light-tree cannot be changed until the transmission going through the tree is finished; hence it can only serve new requests whose destinations are supersets of the destinations of the established light-tree. The first set of four light-tree methods were proposed in [14], where an established light-tree can only be utilized to groom requests with the same destination set (though not necessarily the same source). Since the probability that two multicast sessions have the same destination set is low, the algorithms may lead to low network resource utilizations.

To release the "same destination set" constraint, algorithms were proposed to allow a new multicast session to utilize multiple existing light-trees if, and only if, these trees have disjoint destination sets which are all subsets of the destinations of the new session [15, 16]. Such algorithms include Multicast Tree Decompose (MTD) [15], LTD-DBNG and LTD-ANCG [16] etc. They adopted the same objective of utilizing the existing light-trees to serve as many destinations as possible. Compared to MTD, LTD-DBNG and LTD-ANCG have an additional step: they split a new light-tree into a few smaller ones such that the smaller trees have a better chance to be used by future multicast sessions. Specifically, while LTD-DBNG splits a light-tree at an intermediate node only if this node is one of the destinations of the multicast session, LTD-ANCG allows more flexible splitting of a light-tree into a number of sub-trees according to a predefined priority list of different sub-tree topologies. Among the existing fixed light-tree methods, LTD-ANCG achieves the best blocking performance; hence it will be adopted in the performance comparisons.

For the adaptive light-tree methods, established light-trees can be changed dynamically for multicast traffic grooming [17, 18], though this may cause interruptions to ongoing traffic. Specifically, by modeling the network into a layered auxiliary graph, the MDTGA algorithm allows its established light-trees to be dropped, branched or extended to groom new requests, or to be contracted to release unused resources [17]; the EMGA method adopts the similar mechanism, but it dynamically changes its link costs to improve the overall grooming efficiency [18]. It is shown in [18] that EMGA achieves better blocking performance than MDTGA in different cases. In this paper, we include EMGA in performance comparisons.

An interesting stop and go (S/G) light-tree mechanism was proposed in [19]. However, since it is based on a very different hybrid circuit- and packet-switched architecture, we will not involve it in comparisons.

III. Lightpath vs. Light-tree: A Brief Comparison

A. Physical Layer Node Architectures

To support multicasting, some or all network nodes need to be able to copy data, either in electronic domain or in optical domain or in both, from a single input port to multiple output ports. To support traffic grooming, a node should be able to convert optical signals into electronic domain, performing appropriate traffic aggregation, and finally convert the signals back into optical domain. The devices that perform the traffic grooming operations are called grooming fabrics [14].

For lightpath based dynamic multicast traffic grooming, network nodes typically have the architecture as shown in Fig. 1(a), which is known as grooming capable optical cross-connect (GC-OXC) [5]. A GC-OXC mainly consists of a wavelength switch fabric and a grooming fabric. While the wavelength switch fabric supports the optical domain data switching operations, the grooming fabric realizes the data duplicating, grooming and switching in the electronic domain. For light-tree based multicast traffic grooming, the multicast capable optical grooming switch (MC-OGSW) architecture [14] as shown in Fig. 1(b) is usually adopted. Specifically, MC-OGSW is also equipped with a grooming fabric responsible for traffic aggregation, re-transmission and termination in electronic domain; while traffic duplication, however, is handled by the light-splitter banks in the optical domain. Note that MC-OGSW may cause some requests to be blocked if it is not equipped with sufficient light-splitter banks. In this paper, since our main focus is to conduct fair comparisons between lightpath and light-tree based schemes, we assume that all MC-OGSWs are equipped with abundant light-splitters banks and hence there is no request blocking due to limited number of light-splitters.

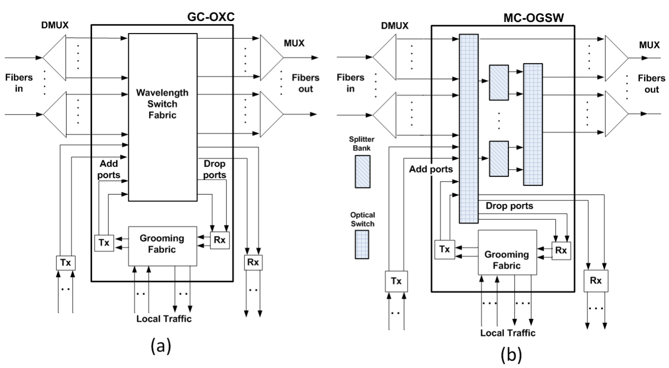


Fig. 1 Node architectures for multicast traffic grooming: (a) GC-OXC for lightpath based schemes. (b) MC-OGSW for light-tree based schemes.

For both architectures, the OXC is equipped with a number of add/drop ports, and the number of ports generally equals the number of transceivers on the node. Note that both the add/drop ports and the transmitters/receivers are of high costs for their high-speed processing units. Hence, to save network cost without sacrificing network performance, each network node is usually equipped with a limited number of such port pairs shared by all wavelengths going through it. In this paper, we define *add/drop ratio* r ($0 < r \leq 1$) as the ratio of the number of add/drop port pairs over the total number of going-through wavelengths. For a node with $r < 1$, we term it a port-limited one; otherwise, we term it a port-unlimited node.

Although previous results claimed that traffic duplication in optical domain using passive light-splitters is less expensive than that in electronic domain [6], using light-splitters does not allow convenient traffic grooming. Moreover, optical splitting may result in higher power losses and degraded signal quality. It is also worth noting that if both architectures are equipped with the same number of transceivers, an MC-OGSW switch may be more complex and expensive than a GC-OXC switch due to the power loss compensation units it may need to have.

B. Network Layer Multicast Traffic Grooming Methods

Existing results have shown that light-tree based schemes consume fewer network resources than lightpath based schemes when serving multicast requests with wavelength level granularity [20]. It remains, however, largely unknown which scheme is a better choice for requests with sub-wavelength granularities. We argue (and confirmed by extensive simulation results as reported later in this paper) that lightpath based methods may have a winning margin in supporting dynamic traffic grooming, mainly because it allows more efficient traffic grooming. An illustrative example is shown in Fig. 2.

In Fig. 2, we assume that each node is equipped with only one pair of transceivers, and the fiber link is bidirectional carrying only a single wavelength in each direction. Suppose a multicast request R1: {S, {C, D}, ¼} arrives at the network, where S, {C, D} and ¼ are the source, destination set and the bandwidth requested versus the channel capacity, respectively.

When the light-tree scheme is utilized, we can easily find a route for it as shown in Fig. 2(a). However, though such a route is optimal for the current request, bandwidth may be wasted when a new request, e.g., R2: {S, {C}, ¼}, arrives. Specifically, by accommodating it using the adaptive light-tree method, e.g., MDTGA or EMGA, R2 will be groomed into R1, and the bandwidth from B to D may be wasted; if a fixed light-tree algorithm is adopted, R2 will be blocked (Note that LTD-ANCG decomposes R1's light-tree as shown in Fig. 2(b).).

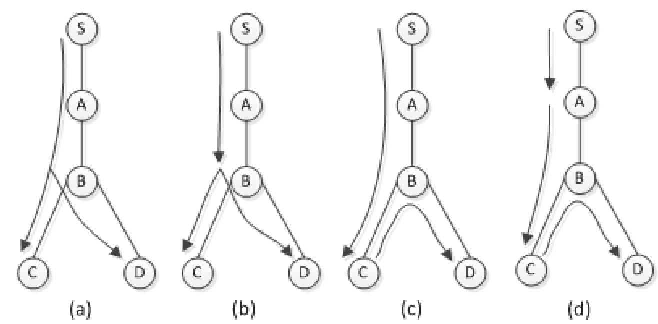


Fig. 2 An example of multicast traffic grooming: (a) ordinary light-tree solution; (b) light-tree decomposition with LTD-ANCG; (c) ordinary lightpath solution; (d) fragmentation of long lightpaths.

When a lightpath based scheme is adopted, however, the situation is different. For R1, a route as shown in Fig. 2(c) can be found; while for R2, it can be easily served by the existing lightpath from S to C. Compared to the ordinary light-tree solution as shown in Fig. 2(a), the lightpath solution achieves a better bandwidth blocking performance at the cost of consuming one more transmitter, but it also saves a light-splitter; while compared to LTD-ANCG as shown in Fig. 2(b), it saves one receiver and one light-splitter.

From the above example, we observe that although lightpath may consume more transceivers for the current request in traffic grooming process, it increases the chance that these transceivers may be conveniently utilized to groom future requests and consequently, helps improve network blocking performance. For networks with given link capacity/transceiver resources, however, it is not easy to tell which scheme performs better. As aforementioned, there are no systematical comparisons between the bandwidth blocking performances of these two types of schemes to the best of our knowledge.

C. Possible Further Improvements

The example above reveals the main advantage of lightpath based schemes that they allow more convenient "sharing" of resources between different multicast sessions. To make further improvements, the chance that different multicast sessions can be groomed should be further increased. One possible approach is to fragment a long lightpath into a few shorter ones if, and only if, such will *not* over-utilize transceiver resources, making them scarcer resources restricting the network performance in handling future multicast sessions.

An example is shown in Fig. 2(c). If a new request R3: {A, {C, D}, ¼} arrives after R1 and R2 have been successfully accommodated. Since there is no available wavelength along the route from A to C, this new request will be blocked, even if node A still has idle transmitters and the residual capacities along all the links are also sufficient. However, if we fragment the long lightpath from S to C at the intermediate node A at the moment when it is set up, as shown in Fig. 2(d), then R3 can be provisioned. Compared to the ordinary lightpath solution, the fragmentation process consumes an additional pair of transceivers when serving R1, yet it prevents R3 from being blocked, and may further help support some future requests initiating or terminating at node A.

The above example shows that, by properly fragmenting long lightpaths into a few shorter ones while carefully keeping a balance between saving link capacity resources and transceiver resources, the network blocking performance may be improved.

IV. Lightpath Fragmentation (LPF) Algorithm for Dynamic Multicast Traffic Grooming

A. Problem Statement and Main Idea

Let a network be represented as a graph *G(V, E)*, where *V* denotes the set of network nodes and *E* the fiber links. Assume that each link is composed of two fibers in opposite directions, each carrying *W* wavelengths. A multicast request is represented as *R{s, D, b}*, where *s*, *D* and *b* denote the source, the set of destination nodes, and the required bandwidth of the request, respectively. A request is served only when all its destination nodes can be served; otherwise, the request is blocked.

The dynamic multicast traffic grooming problem can be defined as follows. In a given network with dynamic arrivals of unicast/multicast connection requests, based on the global information of link state and availability of transmitter/receiver resources on each node, a centralized algorithm is to be devised to support these connection requests with the objective of minimizing the network BBR. Note that we assume that any ongoing transmission cannot be interrupted. In other words, an existing lightpath cannot be fragmented or rerouted when there is ongoing transmission going through it.

In this section, we propose a new lightpath-based multicast traffic grooming algorithm which we term as LightPath Fragmentation (LPF) algorithm. The main idea of the algorithm is to enhance the resource sharing between different multicast sessions by adopting proper fragmentation of long lightpaths. We shall show that, by keeping a balance between link capacity resources and transceiver resources, LPF algorithm significantly outperforms all the existing algorithms.

Below we shall first discuss how to choose the nodes along a lightpath for lightpath fragmentation and then present a detailed description of the LPF algorithm.

B. Selection of Fragmentation Nodes

Upon the arrival of a new multicast request, a tree route will be found for it. When lighpath based schemes are adopted, each tree may contain one or more lightpaths, each of which traversing one or multiple optical links. As discussed earlier, having too long lightpaths may lower the chance of resources sharing in supporting future multicast sessions. We propose to fragment newly set-up long lightpaths if, and only if, such will not over-utilize network transceiver resources. Below we discuss such fragmentation process in detail.

Suppose P is a new lightpath that is found for a multicast request, and n_i is an intermediate node of *P*, of which the fanout degree is d_i. Denote the numbers of idle transmitters and receivers on n_i as T_i and R_i respectively, and the numbers of free wavelengths on the incoming and outgoing links which the new lightpath goes through as ω_{in} and ω_{out} respectively. To determine whether P should be fragmented at n_i, the main idea is to figure out whether wavelength channels or transceivers are more limited resources (hereafter termed as *bottleneck* resources) on this node. Lightpath fragmentation happens on node n_i if and only if wavelength channels turn out to be the bottleneck resources. Strictly speaking, which resources are the bottleneck resources may depend on *traffic pattern* which measures the ratio of traffic loads between different source-destination pairs. To avoid complicated calculations relying on the knowledge of traffic pattern, which anyway may not be easily available in many real-life systems, we consider the benchmark case where a lightpath is to be set up from node n_i to each of the other nodes in the network and see which resources becomes the bottleneck first. As later we will see, this simple method steadily leads to satisfactory performance. Specifically, to figure out the bottleneck resources, two parameters are defined as follows:

$$\alpha_m = \min\left(\frac{T_i}{d_i \times \omega_{out}}, \frac{R_i}{d_i \times \omega_{in}}\right) \tag{1}$$

$$\alpha = \frac{1}{H_i} \ , \tag{2}$$

where H_i is the average hop length of the shortest paths from n_i to all the other nodes. In the above equations, α_m denotes the smaller one between the add and drop ratios on n_i, and α is the add/drop ratio required for n_i to support lightpaths from itself to each of the other nodes. Note that for a certain network node, α is a constant once the network topology is given.

For each new lightpath that is found for supporting a multicast request, α_m is calculated on each intermediate node along the lightpath and then compared to α of the node. If $\alpha_m > \alpha$, transceivers are not regarded as bottleneck resources and hence the lightpath is fragmented at this node; whereas when $\alpha_m \le \alpha$, it indicates that the remaining transceiver resources on this node are limited and saving the transceiver resources may help handle future requests. The lightpath is therefore not fragmented at this node. Adopting this simple strategy, we propose the LPF algorithm for dynamic multicast traffic grooming. Note that the algorithm merely aims at optimizing network blocking performance without worrying any possible drawbacks of lightpath fragmentation, e.g., the possible longer delay. The algorithm however can be revised to take into account some other constraints, as later will be briefly discussed in Section V-D.

Below we present the LPF algorithm in detail.

C. LPF Dynamic Multicast Grooming Algorithm

Since finding the optimal route for multicast traffic grooming is an NP-complete problem [5], we adopt the simple minimum cost path heuristic (MPH) [21] to find the tree route for each request. The main idea of MPH is to use the minimum cost paths to connect request destinations one by one to the tree for a request.

The main working steps of LPF are shown as follows.

LPF Multicast Traffic Grooming Algorithm

Input:
A network *G*(*V*, *E*), and a multicast request *R{s, D, b}*.
Output:
A set of lightpaths for serving *R{s, D, b}*.
Algorithm:
BEGIN
1. Grooming with existing lightpaths: Generate an auxiliary graph (AG) using existing lightpaths with enough residual bandwidth; Call **MPH** on AG to initiate a tree from *s* to connect as many members of *D* as possible. Remove served members from *D*, and if *D* is empty, go to Step 12; otherwise, save the partial tree *T*, and continue. *//logical-layer grooming*
2. Add *s* and all the nodes on *T* to a set *S*; *//optical-layer processing (2-10)*
3. **While** $D \neq \Phi$ **do**
4. Call **Optical-layer-routing-sub-algorithm** (*G(V, E), S, D*), returns a new lightpath *P*;
5. Call **Lightpath-fragmentation-sub-algorithm** (*G(V, E), S, P*), returns some new lightpaths;
6. **For** each new lightpath P_i *// grooming of fragmented lightpaths*
7. Check existing lightpaths with enough residual bandwidth
8. **If** a certain existing lightpath P_e has the same source and destination as P_i, add P_e onto the partial tree *T* found in Step 1 and delete P_i from the new lightpath set.
9. **End For**
10. **End While**
11. Allocate transceivers and wavelength to each new lightpath. *//resource allocation (11-12)*
12. Update residual capacities of all links of the logical-tree *T*.

END

Procedure: Optical-layer-routing-sub-algorithm

Input:
A network *G*(*V*, *E*), two node sets *S* and *D*.
Output:
A new lightpath *P* to serve a request destination.
Algorithm:
BEGIN
1. Generate an optical-layer AG; calculate all-to-all shortest paths between any node in *S* to any node in *D* by adopting a proper path cost definition (as later defined by Eq. (3)).
2. Choose the shortest one among the shortest paths connecting a certain member in *S* to a certain member *d* in *D*. Denote the distance of the path as *dis*.
3. **If** $dis < \infty$
4. $S = S \cup \{d\}$; $D = D \setminus d$; Save the shortest path *P*;
5. **Else**
6. Block the request *R*, **break**;

END

Procedure: Lightpath-fragmentation-sub-algorithm

Input:
A network *G*(*V*, *E*), a node set *S* and a lightpath *P*
Output:
A set of new lightpaths.
Algorithm:
BEGIN
1. **While** (any node of *P* has not been checked) **do**
2. **For** each intermediate node (if any) n_i along *P*
3. Calculate α_m for n_i
4. **If** $\alpha_m > \alpha$ at n_i
5. Fragment *P* at n_i, and get two new lightpaths P_a and P_b;
6. $S = S \cup \{n_i\}$; $P = P_b$;
7. **End For**
8. **End While**

END

With MPH, LPF utilizes existing logical links to serve as many destinations as possible in Step 1. Steps 2-10 then serve the remaining destinations, if any, by setting up new lightpaths and fragmenting the new lightpaths when applicable. Specifically, Step 4 serves a destination node by setting up a new lightpath; Step 5 fragments this new lightpath when applicable; and Steps 6-9 try to groom the fragmented new lightpaths into existing logical links. Finally, Steps 11-12 update the network status. Note that on both the logical and the optical layers, the "first-fit" wavelength assignment policy is adopted.

Different path length definitions can be adopted in Step 4. The simplest way is to define the path cost as equaling its hop length. In our experiences, even by adopting this simple definition, LPF manages to outperform all the existing methods, in many cases by more than an order of magnitude. To further improve the performance, a better path cost definition has been proposed in [22] as follows:

$$C_{ij} = \begin{cases} \dfrac{(1-r)}{p \times r(\overline{H}+1)} - H_{ij} \ln\left(1 - \dfrac{1}{\omega_{ij}+1}\right) & \text{if } \omega_{ij}>0 \text{ and } p>0 \\ \infty & \text{if } \omega_{ij}=0 \text{ or } p=0 \end{cases} \qquad (3)$$

where p is the smaller one among the number of transmitters at the source and the number of receivers at the destination of the lightpath; ω_{ij} is the number of available wavelengths along the lightpath route; $\overline{H}$ is the average path length of the network, and H_{ij} is the minimum hop length between two end-nodes of the lightpath. This function helps keep balanced consumptions of wavelength and transceiver resources. Specifically, if both resources are abundant, the costs of consuming them should be low and not so different from each other; while if any one of them becomes scarce, the cost of consuming the scarce resource becomes higher to impose a penalty to utilizing it. Extensive simulation results have shown that adopting this definition steadily leads to best performance among all the link-cost definitions we have tested. In this paper, we adopt this definition in calculating path cost.

V. Lightpath vs Light-tree: Simulation Results and Discussions

Simulations are carried out to compare the lighptah and light-tree based grooming algorithms in different cases. Below we firstly present the performance metrics and simulation environment, and then show the simulation results in different cases.

A. *Performance Metrics and Simulation Environment*

As mentioned in Section I, performances of all the algorithms are mainly measured by their BBR. To assess the grooming efficiency of lightpath and light-tree, the *average channel capacity utilization* (U_w) is defined as follows,

$$U_w \overset{def}{=} \lim_{T\to\infty} \frac{1}{T}\int_0^T \frac{1}{W_u}\sum_{W_u}\frac{B(t)}{B}dt\,,$$

where $B(t)/B$ measures the portion of channel capacity being utilized and W_u is the number of wavelengths being used. We see that U_w measures the capacity utilization of the channels being utilized for transmission.

To evaluate the required OEO conversions for each request which, as later will be discussed in more details, helps reflect the average intermediate node processing delay experienced by each admitted session, the *average number of OEO conversions per session* is defined as

$$\mathrm{N_{OEO}} = \frac{\sum_{NReq} \text{no. of OEO used for the request}}{NReq}\,,$$

where *NReq* is the number of admitted connection requests.

Two typical network topologies as illustrated in Fig. 3 are adopted in our simulations, which are 14-node, 21-link NSFnet, and 11-node, 26-link COST239 network, respectively. Results shown in each of the following figures are an average of at least five independent simulations, each of which running at least 10^5 connection requests.

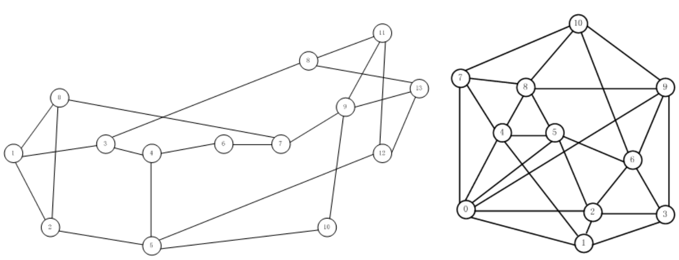


Fig. 3 Two network topologies utilized for simulations: (a) 14-node NSFnet; (b) 11-node COST239.

The following are some assumptions adopted in simulations:

1) For lightpath based algorithms, all network nodes are equipped with GC-OXC, while for light-tree based ones, MC-OGSWs are utilized; for fair comparisons, both switches are equipped with equal number of transceivers. As aforementioned in Section III-A, we assume that all MC-OGSWs have sufficient light-splitters.

2) Each link is composed of two fibers of opposite directions, each carrying $W=32$ wavelengths; the capacity of each wavelength is $B=16$ units.

3) Requests arrive/leave at the network according to a Poisson process with a rate λ, and their holding time follows the negative exponential distribution with a mean of $\mu=1$; the bandwidth for supporting each request is an integer uniformly distributed in [1, 16].

4) The number of transceivers on a node is set to be $W \times d_i \times r$, where d_i is the fanout degree of the node.

5) Signal power loss due to light-splitting or transmission attenuation is neglected.

Four request generation models are utilized for simulations:

M1: a number of randomly pre-selected requests arrive/leave the network independently. For NSFnet, such requests are {7, {1, 5, 8}}, {3, {9, 12, 6}}, {10, {2, 4, 8}} and {11, {0, 13, 6}}; while for COST239, the requests are {2, {10, 5, 7}}, {1, {0, 8, 5}}, {4, {6, 3, 10}} and {9, {5, 7, 3}}.

M2: the source and destination sets of a request are randomly chosen from two separate pre-selected sets. Similar to that in M1, we define such two sets as {7, 3, 10, 11} and {{1, 5, 8}, {9, 12, 6}, {2, 4, 8}, {0, 13, 6}} for NSFnet, and {2, 1, 4, 9} and {{10, 5, 7}, {0, 8, 5}, {6, 3, 10}, {5, 7, 3}} for COST239.

M3: the request source and destinations are randomly chosen among all network nodes, with the destination number being limited to a certain range. In this paper, the destination number is uniformly distributed in [2, 4] for both topologies.

M4: request source and destinations are randomly chosen; the destination number is distributed in [2, *N*-1] following the truncated geometric distribution with parameter q [16]. The average destination number is set to be 3, and thus, for NSFnet, $q=0.501$, while for COST239, $q=0.504$.

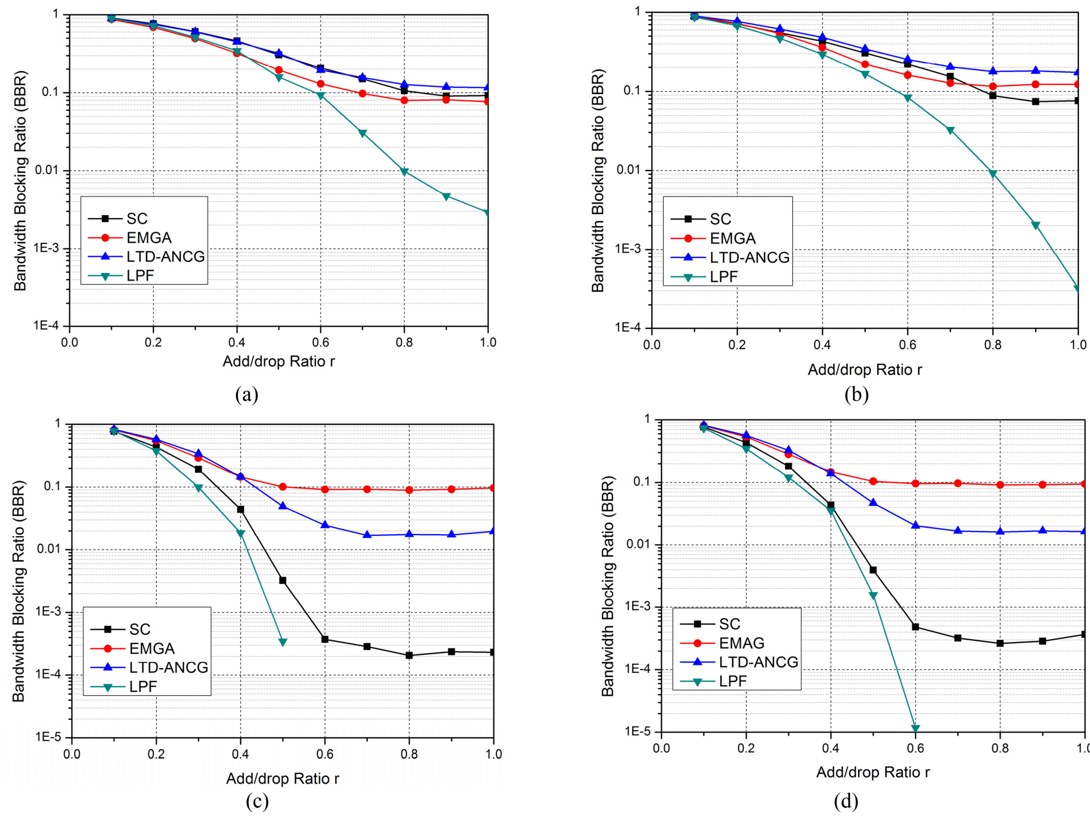


Fig. 4. (Color online) Lightpath vs light-tree: BBR performances of the four different methods vs. add/drop ratio under traffic with four different patterns. Traffic load is set to be 250 Erlangs. (a) M1. (b) M2. (c) M3. (d) M4.

By testing on the above models, we shall be able to evaluate the performance of algorithms in different cases with uniform and non-uniform traffic patterns and relatively larger or smaller numbers of destinations each session, respectively.

Since all the conclusions hold for both topologies, unless otherwise specified, we present only the results on NSFnet for comparisons and discussions.

B. Lightpath vs. Light-tree: Effects of Different Traffic Models and Traffic Loads

Figure 4 compares the BBR performances of different algorithms under a fixed traffic load ρ=250 Erlangs with different traffic patterns. From the simulation results, it is interesting to observe that: 1) for M1 and M2, there is no obvious winner among the *existing* lightpath and light-tree based algorithms, while for M3 and M4, the lightpath based methods have an obvious winning margin over the light-tree based ones within the whole range of add/drop ratio; 2) the LPF algorithm steadily outperforms all the existing algorithms in different cases, esp. when the network has a high add/drop ratio.

It is not difficult to understand why performances of all the existing methods are similar for M1 and M2: when there are very limited options in selecting source/destinations of each session, connections are setup and torn down frequently only along a few tree routes. The severely unbalanced distribution of traffic loads in the network degrades network resource utilizations and causes high BBR for all the algorithms. When multicast sessions are more evenly distributed in the network, e.g., as that in M3 and M4, however, those algorithms helping achieve higher network resource utilizations easily gain a nontrivial winning margin.

Note that even for M1 and M2, the LPF method steadily outperforms all the existing methods, esp. in networks with a high add/drop ratio, thanks to its enhanced resource sharing.

Due to space limit, hereafter we adopt M3 as a representative case for performance comparisons, since the request generation in M1 and M2 has too limited flexibility, while M4 may lead to some over-sized light-trees (and rather bad performance) when using the light-tree methods. Note that the conclusions always hold that i) the differences between the performances of different algorithms tend to be more significant under M3 and M4 compared to those under M1 and M2; and ii) LPF always outperforms all the existing algorithms.

Figure 5 compares the performances of different algorithms under a higher traffic load of 450 Erlangs. As can be seen, LPF remains as the best-performing algorithm. In fact, it is the only algorithm which manages to drive BBR to be ever lower than 1%. With a high add/drop ratio, it can drive BBR all the way to be lower than 0.01%.

The observation that LPF outperforms the even best existing lightpath based method, namely the SC method, by more than an order of magnitude when add/drop ratio $r \geq 0.7$ showcases the benefits of lightpath fragmentation: the better utilization of the redundant transceiver resources helps enhance wavelength capacity sharing as well as releasing the wavelength continuity constraint, and consequently, significantly improve the network blocking performance.

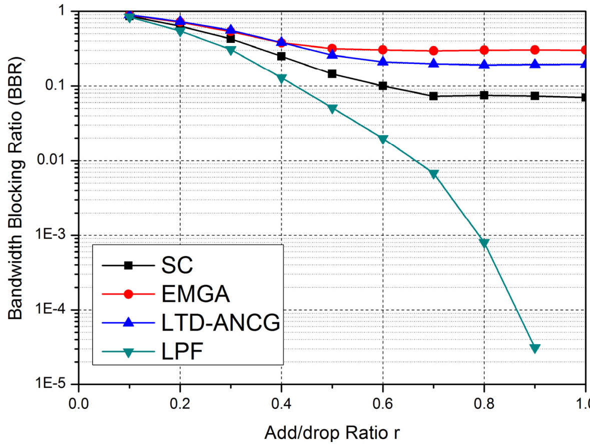


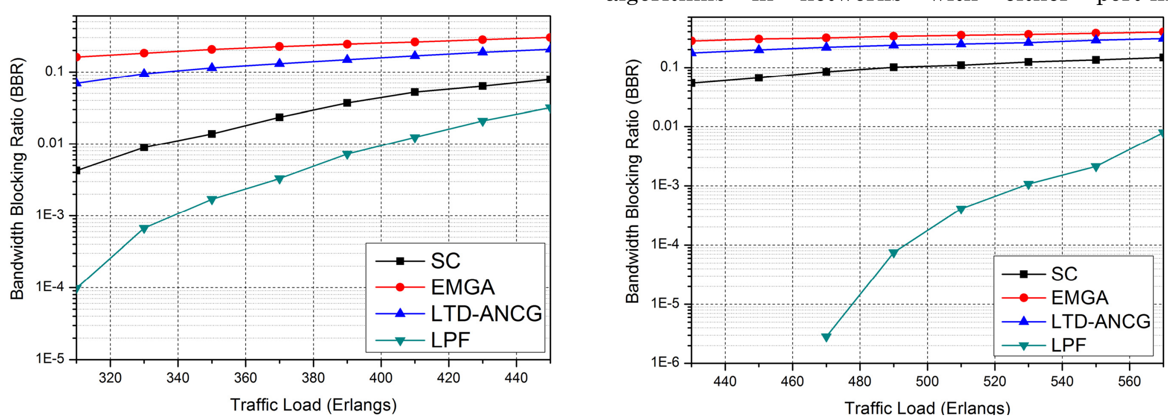
Fig. 5 (Color online) Lightpath vs light-tree: BBR performance versus add/drop ratio *r* under high traffic load ρ=450Erlangs for M3.

Figure 6 further compares the BBR performances of different algorithms with given add/drop ratios but under varying traffic loads. Specifically, we consider two different cases where $r = 0.6$ and $r = 1.0$, respectively. As can be seen, though the differences between BBR of the LPF algorithm and those of the other algorithms tend to decrease under increasing traffic loads (which is not a surprise), even when under very heavy traffic loads where BBRs of all the other algorithms are close to or higher than 10%, LPF algorithm still outperforms all the other algorithms by more than 50% where $r = 0.6$. For the case where $r = 1.0$, the difference is always higher than an order of magnitude; under moderate traffic loads, the differences are in three or four orders of magnitude.

C. Lightpath vs. Light-tree: Effects of the Number of Multicast Destinations and the Fraction of Unicast Traffic Loads

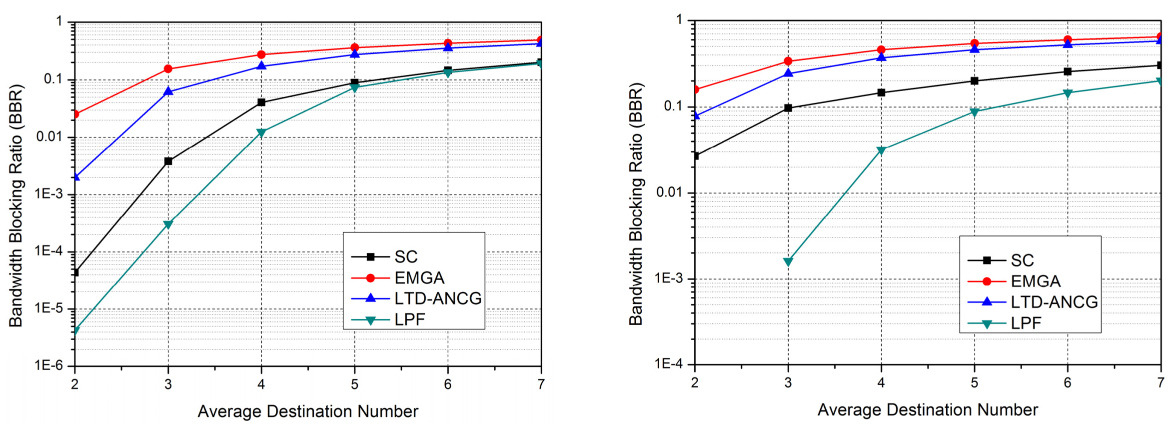
In this section, we evaluate how the BBR performances of all the algorithms change with two important factors of traffic pattern, namely the number of multicast destinations and the fraction of unicast traffic loads vs. the overall traffic loads. Specifically, for the former evaluation, we adopt a request generating model that is similar to M3 but with the number of destinations uniformly distributed in [2, $2(D_n - 1)$], where D_n is the average number of destinations per multicast session; while for the latter evaluation, we define *unicast traffic ratio* as the ratio of unicast traffic to the overall network traffic loads, and let the destination number of each multicast request be uniformly distributed in [2, 4].

Figure 7 shows the BBR performances of all the algorithms in networks with either port-limited or



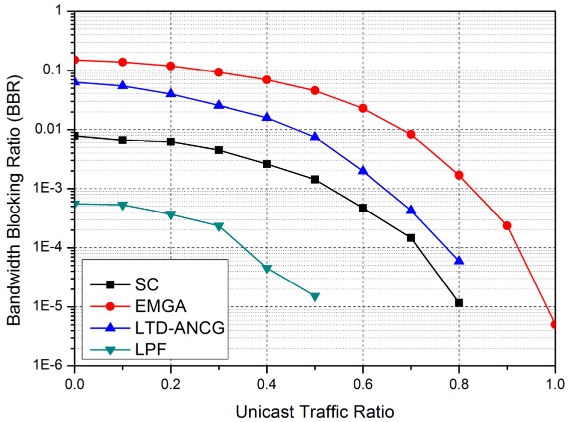
(a) (b)

Fig. 6. (Color online) Lightpath vs Light-tree: BBR under different traffic loads. (a) Network with port-limited nodes where *r*=0.6. (b) network with port-unlimited nodes where *r*=1.0.

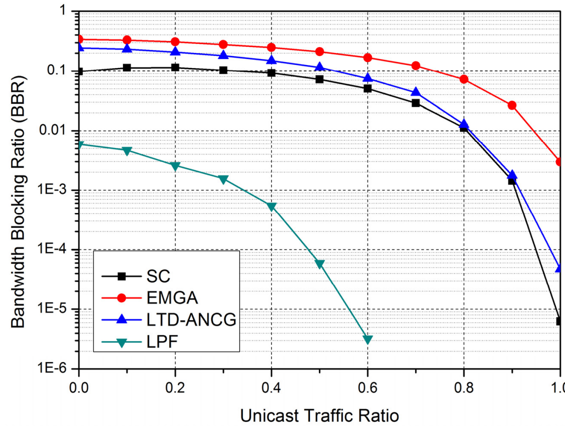


(a) (b)

Fig. 7. (Color online) Lightpath vs Light-tree: effects of the number of multicast destinations. (a) Network with port-limited nodes where r=0.6, traffic loads are 300 Erlangs. (b) Network with port-unlimited nodes where r=1.0, traffic loads are 500 Erlangs.



(a)



(b)

Fig. 8. (Color online) Lightpath vs Light-tree: effects of the unicast traffic ratio. (a) Network with port-limited nodes where r=0.6, traffic loads are 300 Erlangs. (b) Network with port-unlimited nodes where r=1.0, traffic loads are 500 Erlangs.

port-unlimited nodes. As can be seen, under a fixed traffic load, the BBR performances of all the algorithms degrade with an increasing average number of destinations. The main reason is obvious: requests with larger number destinations tend to consume more network resources. And it is not a surprise to see that LPF has a larger winning margin in a port-unlimited network than that in a port-limited network since it makes better use of the redundant transceiver resources.

Figure 8 compares BBR performances of different algorithms with an increasing unicast traffic ratio, still in networks with either port-limited or port-unlimited nodes. Simulation results show that in both networks, the BBR performances of all the algorithms improve with an increasing unicast traffic ratio, and again LPF has a bigger winning margin over the other algorithms in the port-unlimited networks than that in port-limited networks.

Figures 7 and 8 also show that the lightpath based algorithms steadily outperform the light-tree based ones, and LPF outperforms its closest competitor, the SC algorithm, by at least one order of magnitude under most cases. The only exception is when the average number of multicast destinations is very large, where the exhausted link capacity resources drive all the algorithms to have a high BBR. In the most favorable case where the traffic loads are mostly unicast traffic, LPF may outperforms its closest competitor by more than three orders of magnitude.

D. Lightpath Based Schemes: Capacity Utilization, Delay and Consumptions of Transceiver Resources

Figure 9 compares the average capacity utilization of non-idle wavelength channels of different algorithms. As can be seen, lightpath based schemes achieve much higher utilizations than light-tree based ones within the whole range of add/drop ratio. Such an observation is not a surprise since, as mentioned earlier, lightpath based schemes help achieve much easier grooming of traffic loads and consequently, much higher utilization of wavelength capacity resources.

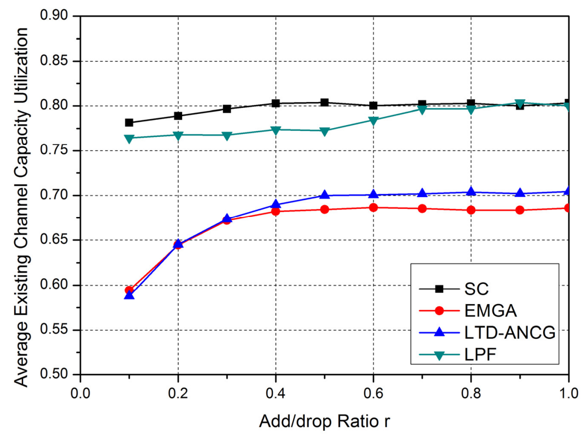


Fig. 9 (Color online) Lightpath vs light-tree: average existing-channel capacity utilization

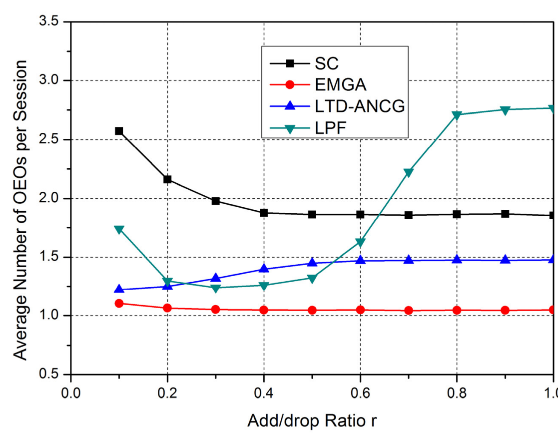


Fig. 10 (Color online) Average number of OEO conversions experienced by each admitted request. Traffic loads are 250 Erlangs.

While lightpath based algorithms steadily lead to better BBR performance, a natural concern is that they may lead to more OEO conversions for each connection request and consequently leading to a longer delay. Figure 10 compares the average number of intermediate OEO conversions each connection has to go through, which provides an indirect yet good measurement of the expected delay experienced by each admitted request. As we can see, for $r \le 0.4$, the numbers of

OEO conversions required by SC and LPF decrease with an increasing value of r; while when $r > 0.4$, such numbers increase with r for LPF, yet stay largely unchanged for SC. For EMGA and LTD-ANCG, such numbers remain largely unchanged throughout the whole range of r.

Such observation can be understood: when transceiver resources are too limited, e.g., $r = 0.1$, the lightpath based methods can only set up a small number of lightpaths. Consequently, any admitted request tends to be groomed into existing lightpaths as far as such is possible, which causes a relatively large number of OEO conversions per request. When transceiver resources become more redundant, more lightpaths can be set up, many of which may bypass some intermediate nodes along its route. The number of intermediate OEO conversions therefore decreases. When the number of transceivers further increases, e.g., $r > 0.4$, however, LPF starts to fragment more new lightpaths to improve link capacity utilizations, which pushes up the number of OEO conversions per request. For SC, on the other hand, it will not increase the transceiver consumptions per connection request even if such resources are redundant; thus its OEO conversions stay largely unchanged once it reaches its lowest value. For the light-tree based methods, since traffic grooming happens at a much lower probability, the number of OEO conversions per admitted request remains rather insensitive to the redundancy of the transceiver resources.

Figure 10 also reveals that while LPF may introduce a higher average number of OEO conversions per admitted request when $r > 0.65$, it significantly outperforms the other methods by using roughly the same, or even fewer, OEO conversions per admitted request when $r < 0.65$. For example, putting Fig. 4(c) and Fig. 10 together, we see that when $r = 0.5$, LPF outperforms both SC and LTD-ANCG with fewer OEO conversions per admitted request.

Figure 11 further shows that, compared to SC, LPF in fact does *not* consume more transceiver resources. Specifically, we compare the average number of transceiver pairs being utilized in the whole network within a period of time (Note that we do not compare the transceiver consumptions between lightpath based and light-tree based methods since a light-splitter in a light tree functionally acts as a one-to-many transceiver which, however, may request other resources such as more optical amplifiers.). As we can see, the total number of transceiver pairs consumed by LPF is actually slightly lower than that by SC within the whole range of r: the higher number of OEO conversions per admitted request does not necessarily means a higher overall consumption of transceiver resources. The fragmentation of lightpath helps increase the efficiency of traffic grooming, and consequently keeps overall transceiver resource consumptions at a reasonably low level.

Note that in this paper, we define the sole objective of LPF as improving the network BBR performance, without worrying about any possible drawbacks such as longer processing delay or higher transceiver consumptions etc. As a result, the algorithm tends to make full use of the transceiver resources when they are redundant. The algorithm can be easily revised to keep a balance between blocking performance and resource consumptions. For example, instead of measuring whether it helps improve network blocking performance by fragmenting a lightpath on each of its intermediate nodes, we may revise the scheme to limit the number of fragmentations we could have for each lightpath (The upper bound of fragmentations we could have for a lightpath may depend on its hop length, the redundancy of its capacity resources, and/or something else.) by fragmenting only on a few "most favorable" nodes along each lightpath. Such possible extensions, however, are out of scope of this paper and have to be discussed in our future work.

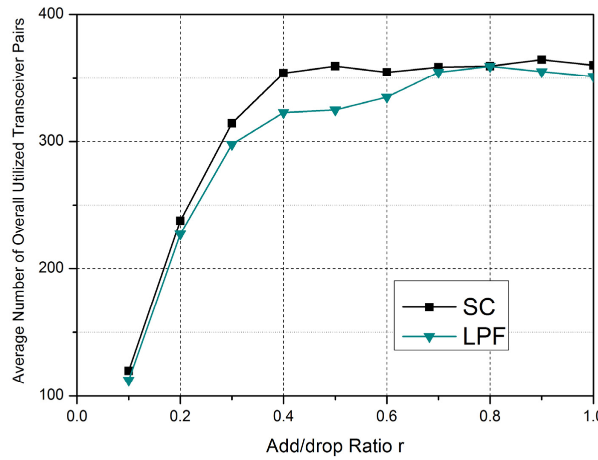


Fig. 11 (Color online) Average number of transceiver pairs utilized all over the network, counted from the arrival of the 5,000-th connection request to the 10^5-th connection request. Traffic loads are 250 Erlangs.

VI. Conclusion

In this paper, we studied dynamic multicast traffic grooming in meshed WDM networks. Extensive simulations were carried out to compare various existing lightpath and light-tree based traffic grooming algorithms. We found that the lightpath based methods steadily enjoy a winning margin in network bandwidth blocking performance over the light-tree based ones. Inspired by such observations, we proposed the LightPath Fragmentation (LPF) method to make further improvements. Simulation results showed that LPF steadily outperforms all the existing methods, mostly by at least one order of magnitude, with a moderate cost of longer delay. The transceiver resource consumptions meanwhile remain as comparable with those of the best existing lightpath based method. Effects of different factors including the average number of multicast destinations and the unicast traffic ratio were also evaluated.

Acknowledgment

The authors thank Mr. Rongping Lin and Dr. Bin Chen for the helpful discussions and for sharing with the authors the source codes of their algorithms.

References

[1] R. Ramaswami, "Optical networking technologies: what worked and what didn't," *IEEE Commun. Mag.*, vol. 44, no. 9, pp. 132–139, Sep. 2006.

[2] I. Chlamtac, A. Ganz and G. Karmi, "Lightpath communications: an approach to high bandwidth optical WAN's," *IEEE Trans. Commun.*, vol. 40, no. 7, pp. 1171-1182, Jul. 1992.

[3] G. N. Rouskas and H. G. Perros, "A tutorial on optical networks," *NETWORKING 2002 Tutorials, LNCS 2497*, pp. 155-193, May 2002.

[4] L. H.Sahasrabuddhe and B. Mukherjee, "Light-trees: optical multicasting for improved performance in wavelength-routed networks," *IEEE Commun. Mag.,* vol. 37, no.2, pp. 67-73, Feb. 1999.

[5] A. E. Kamal, "Algorithms for multicast traffic grooming in WDM mesh networks," *IEEE Commun. Mag.*, vol. 44, no. 11, pp. 97-105, Nov. 2006.

[6] H. V. Madhyastha, G. V. Chowdhary, N. Srinivas and C. S. R. Murthy, "Grooming of multicast sessions in metropolitan WDM ring networks," *Computer Networks*, vol. 49, no. 4, pp. 561-579, Nov. 2005.

[7] N. K. Singhal, L. H.Sahasrabuddhe and B. Mukherjee, "Optimal multicasting of multiple light-trees of different bandwidth granularities in a WDM mesh network with sparse splitting capabilities," *IEEE/ACM Trans. Netw.*, vol. 14, no. 5, pp. 1104-1117, Oct. 2006.

[8] R. Ul-Mustafa and A. E. Kamal, "Design and provisioning of WDM networks with multicast traffic grooming," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 4, pp. 37-53, Apr. 2006.

[9] Y. Zhu, Y. Jin, W. Sun, W. Guo, W. Hu, W.D. Zhong and M.-Y. Wu, "Multicast flow aggregation in IP over optical networks," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 5, pp. 1011-1021, Jun. 2007

[10] S. S. W. Lee, H. H. Yen and A. Chen, "Traffic grooming for IP multicast over WDM networks using light-path and light-tree schemes," in *Proc. of the Ninth International Conference on Networks ICN 2010*, pp. 291-293, Apr. 2010.

[11] M. A. Saleh and A.E. Kamal, "Dynamic provisioning of optical networks with many-to-many traffic grooming," *In Proc. IEEE ICCCN '2010,* pp. 1-7, Aug. 2010.

[12] G. V. Chowdhary, C. Siva and R. Murthy, "Dynamic multicast traffic engineering in WDM groomed mesh networks," in *Proc. IEEE Broadnets'2004*, 2004.

[13] B. Chen, W. D. Zhong, S. K. Bose and Y. Jin, "Applying saturated cut method in dynamic multicast traffic grooming in IP/MPLS over WDM mesh networks," in *Proc. 5th Int. Conf. on Information, Communication and Signal Processing (ICICS)* '2005, pp. 184-189, 2005.

[14] A. Khalil, A. Hadjiantonis, C. M. Assi, A. Shami, G. Ellinas and M. Ali, "Dynamic provisioning of low-speed unicast/multicast traffic demands in mesh-based WDM optical networks," *IEEE/OSA J. Lightw. Technol.*, vol. 24, no. 2, pp. 681-692, Feb. 2006.

[15] C. Lu, X. Nie, S. Wang and L. Li, "Efficient dynamic multicast traffic grooming algorithm on WDM networks," in *Proc. SPIE Int. Soc. Opt. Eng.*, vol. 6022, pp. 602230-1-602230-10, Nov. 2005.

[16] R. Lin, W. D. Zhong, S. K. Bose and M. Zukerman, "Light-tree configuration for multicast traffic grooming in WDM mesh networks", *Photonic Netw. Commun.*, vol. 20, no. 2, pp. 151-164, Oct. 2010.

[17] X. Huang, F. Farahmand and J. P. Jue, "Multicast traffic grooming in wavelength-routed WDM mesh networks using dynamically changing Light-trees," *IEEE/OSA J. Lightwave Technol.*, vol. 23, no. 10, pp. 3178-3187, Oct. 2005.

[18] L. Liao, H. Wen, L. Li and S. Wang, "New dynamic grooming approaches for multicast traffic in meshed wavelength-division-multiplexing networks," *SPIE Opt. Eng.*, vol. 46, no. 2, pp. 25004-1-25004-7, Feb. 2007.

[19] J. E. Sierra, L. F. Caro, F. Solano, J. L. Marzo, R. Fabregat and Y. Donoso, "Dynamic unicast/multicast traffic grooming using S/G light-tree in WDM networks," in Proc. Intern. Symposium on Performance Evaluation of Computer and Telecommunication System (SPECTS) '2008, pp. 263-268, Jun. 2008.

[20] R. Malli, X. Zhang and C. Qiao, "Benefit of multicasting in all-optical networks," in *SPIE Proc. Conf. All-optical Networking*, pp. 209-220, Nov., 1998.

[21] H. Takahashi and A. Matsuyama, "An approximate solution for the Steiner problem in graphs," *Math. Japonica*, vol. 24, no. 6, pp. 573-577, 1980.

[22] X. Yu, G. Xiao and T. Cheng, "Historical data learning based dynamic LSP routing for overlay IP over WDM networks," submitted to *Optical Fiber Technology*.