Clustering Methods for Hierarchical Traffic Grooming in Large-Scale Mesh WDM Networks

Bensong Chen, George N. Rouskas, and Rudra Dutta

***Abstract*—We consider a hierarchical approach for traffic grooming in large multiwavelength networks of a general topology. Inspired by similar concepts in the airline industry, we decompose the network into clusters, and select a hub node in each cluster to groom traffic originating and terminating locally. At the second level of the hierarchy, the hub nodes form a virtual cluster for the purpose of grooming intracluster traffic. Clustering and hierarchical grooming enables us to cope with large network sizes and facilitates the control and management of traffic and network resources. Yet, determining the size and composition of clusters so as to yield good grooming solutions is a challenging task. We identify the grooming-specific factors affecting the selection of clusters, and we develop a parameterized clustering algorithm that can achieve a desired trade-off among various goals. We also obtain lower bounds on two important objectives in traffic grooming: the number of lightpaths and wavelengths needed to carry the subwavelength traffic. We demonstrate the effectiveness of clustering and hierarchical grooming by presenting the results of experiments on two network topologies that are substantially larger than those considered in previous traffic grooming studies.**

***Index Terms*—Optical networking; Traffic grooming; Network design; Resource provisioning; Hierarchical grooming; Routing; Control plane algorithms; Large networks**.

I. Introduction

Ongoing advances in optical network and communication technologies continue to expand the capacity of individual wavelengths and increase the availability of wavelength channels for direct optical connections. Traffic grooming, the area of research concerned with efficient and cost-effective transport of subwavelength traffic over multigranular networks, has emerged as an important field of study in recent years. In *static* grooming [1], the objective is to provision the network to carry a set of long-term traffic demands while minimizing the overall network cost; the latter is typically taken as the number of electronic ports needed to originate and terminate the set of lightpaths in the logical topology. In *dynamic* grooming [2], on the other hand, the goal is to develop online algorithms for efficiently grooming and routing of connections that arrive in real time. The reader is referred to [3] for a comprehensive survey and classification of research on traffic grooming.

Early work on traffic grooming focused on the ring topology [1,4,5], reflecting the technological push in response to the industry's effort to upgrade the deployed SONET infrastructure to WDM technology. More recently, several studies have begun to address grooming issues in networks with a general topology [6–10]. Nevertheless, most studies regard the network as a flat entity for the purposes of lightpath routing, wavelength assignment, and traffic grooming. In general, such approaches do not scale well to networks of realistic size for two reasons: first, the running-time complexity of traffic grooming algorithms increases rapidly with the size of the network, and second, the operation, management, and control of multigranular networks becomes a challenging issue in large, unstructured topologies.

We have recently proposed a scalable hierarchical framework for traffic grooming that can be applied to networks of practical size covering a national or international geographical area [11]. In our model, the network is organized in a hierarchical manner to facilitate the control and management of resources (e.g., grooming ports and wavelengths) and to ensure the scalability of grooming algorithms and functionality. The model borrows ideas from the hub-and-spoke paradigm used within the airline industry. The network is partitioned into clusters, and one node within

Manuscript received January 21, 2010; accepted April 2, 2010; published July 9, 2010 (Doc. ID 123151).

Bensong Chen (bensong@google.com) is with Google, Inc., Mountain View, California 94043, USA.

George N. Rouskas (rouskas@ncsu.edu) and Rudra Dutta (rdutta@ncsu.edu) are with the Department of Computer Science, North Carolina State University, Raleigh, North Carolina 27695-8206, USA.

Digital Object Identifier 10.1364/JOCN.2.000502

1943-0620/10/080502-13/$15.00

each cluster is selected as the *hub*. Nonhub nodes route all their traffic to the hub, where it is groomed before it is forwarded to the destination cluster; as a result, the hub is the only node in a cluster responsible for grooming traffic not originating or terminating locally. At the second level of the hierarchy, the first-level hubs form another cluster for grooming and routing intercluster traffic. This hierarchical approach is quite scalable, can be used directly for networks with tens or even hundreds of nodes, and is applicable to both static and dynamic grooming contexts.

One important yet challenging issue in the hierarchical grooming approach is the selection of clusters and hub nodes. Although clustering techniques are used in a wide range of network design problems, there is little work related to traffic grooming; we present a survey in Section IV. Therefore, we develop a new parameterized clustering algorithm appropriate for traffic grooming. The algorithm is flexible and allows the network designer to achieve a desired balance among a number of conflicting goals. We also develop lower bounds on two metrics of importance, the number of lightpaths and wavelengths needed for the static grooming problem. To demonstrate the effectiveness of the clustering and hierarchical grooming algorithms, we apply them to two large networks, including a 128-node, 321-link topology corresponding to a worldwide backbone network; the latter is approximately an order of magnitude larger than networks that have been considered in previous grooming studies.

Following the introduction, we describe the hierarchical grooming approach in Sections II and III. In Section IV we discuss clustering methods in general, and in Section V we present and analyze our clustering algorithm for hierarchical grooming in general topologies. We obtain lower bounds in Section VI, and present numerical results on various network topologies and traffic patterns in Section VII. We conclude the paper in Section VIII.

II. Hierarchical Grooming in Mesh Networks

We consider a network of general topology with N nodes. Physical links are bidirectional and support W wavelengths per direction. The capacity C of each wavelength channel is an integer multiple of a basic transmission unit (e.g., OC-3); C is also known as the *grooming factor*. The demands placed on the network are provided in a traffic demand matrix, $T=[t^{(sd)}]$, where integer $t^{(sd)}$ denotes the amount of (forecast) long-term traffic to be carried from node s to node d. Although the traffic demands may change over time, we assume that such changes take place over longer time scales; hence, for the rest of the paper we assume that the traffic matrix is fixed. We also allow the traffic demand between any pair of nodes to exceed the capacity of a wavelength.

The objective of the traffic grooming problem is to configure the network (i.e., determine the lightpaths to be set up) to carry the entire traffic matrix T while minimizing the total number of electronic ports required at the network nodes. Since each lightpath requires exactly two electronic ports (one at the node at each end of the lightpath), this objective is equivalent to minimizing the number of lightpaths in the resulting logical topology. For a more formal definition of the problem and a general-purpose integer linear programming (ILP) formulation, the reader is referred to [3].

The traffic grooming problem in general topology networks has long been known to be NP-hard, since it contains as a subproblem, the lightpath routing and wavelength assignment (RWA) problem, which is itself NP-hard [12]. Furthermore, our earlier work [13,14] has shown that traffic grooming remains intractable even in simple network topologies, such as paths and stars, for which the RWA subproblem can be solved in polynomial time. Consequently, for WDM networks with more than a few nodes, it is important to develop heuristic algorithms that are scalable and can be used to obtain provably good solutions in polynomial time.

Our framework for hierarchical traffic grooming was developed for large-scale mesh networks consisting of several tens or even hundreds of nodes, for which existing grooming algorithms, especially those based on ILP formulations, are not practical. This framework was inspired by the hub-and-spoke paradigm that is widely used by the airline industry. In our approach, a large network is partitioned into a number of clusters, each consisting of a contiguous subset of nodes. The clusters may correspond to independent administrative entities or may be created solely for the purpose of simplifying resource management and control functions.

In the traffic grooming context, we view each cluster as a *virtual star*, and we designate one node as the *hub* of the cluster. We refer to each cluster as a virtual star because, even though the physical topology of the cluster may take any form (and in fact may be quite different than a *physical star* topology), the hub is the only node responsible for grooming intracluster and intercluster traffic. Consequently, hub nodes are expected to be provisioned with more resources (e.g., larger number of electronic ports and higher switching capacity for grooming traffic) than nonhub nodes. Returning to the airline analogy, a hub node is similar in function to airports that serve as major hubs; these airports are typically larger than nonhub airports, in

terms of both the number of gates ("electronic ports") and physical space (for "switching" passengers between gates).

Our hierarchical framework consists of three phases:

1) **Clustering of network nodes**. In this phase, the network is partitioned into m clusters, and one node in each cluster is designated as the hub. The clustering phase is crucial to the quality of the grooming solution. In particular, we have found that the selection of cluster size and hub nodes, as well as the existence of critical cutting edges in the network topology, has a profound effect on the number of lightpaths and wavelengths required for the final solution. We discuss related work on clustering techniques in Section IV, and we describe in detail our clustering algorithm for traffic grooming in Section V.
2) **Hierarchical logical topology design and traffic routing**. The outcome of this phase is a set R of lightpaths for carrying the traffic demand matrix T, and a routing of individual traffic components $t^{(sd)}$ over these lightpaths. This phase is further subdivided into three parts:
   a) setup of direct lightpaths for large traffic demands,
   b) intracluster traffic grooming, and
   c) intercluster traffic grooming.

   This hierarchical approach is discussed in Section III.
3) **Lightpath routing and wavelength assignment (RWA)**. The goal of the RWA phase is to route the lightpaths in R over the physical topology and color them by using the minimum number of wavelengths. The RWA problem on arbitrary network topologies has been studied extensively in the literature [8,9,12,15,16]. In this work, we adopt the LFAP (longest first alternate path) algorithm [15], which is fast, conceptually simple, and has been shown to use a number of wavelengths that is close to the lower bound for a wide range of problem instances.

The outcome of the clustering phase is a partition of the network nodes into some number m of clusters, denoted $B_1, \ldots, B_m$, and the selection of one node, denoted h_i, to serve as the hub for cluster B_i. Consider, for instance, the 32-node network in Fig. 1. The figure shows a partition of the network into eight clusters, $B_1, \ldots, B_8$, each cluster consisting of four nodes. These clusters represent the first level of the hierarchy. Within each cluster, one node is the hub; for instance, node 2 is the hub for cluster B_1. At the second level of the hierarchy, we view the hub nodes of the eight first-level clusters as forming another cluster, and we select one of these nodes as the hub node of this second-level cluster. We emphasize that, while we view each cluster as a virtual star, the actual physical topology of the cluster is determined by the physical topology of the part of the original network where the cluster nodes lie; for example, the four nodes of cluster B_8 form a ring. Since the RWA algorithm is performed on the underlying physical topology *after* the logical topology has been determined, the lightpaths will follow the most efficient paths in the network and are not constrained to go through the hub, as is the case for a physical star. Consider, for example, cluster B_8 with node 32 as its hub. Suppose that the logical topology on the corresponding virtual star with node 32 as the hub includes the one-hop lightpath (28, 32) and the two-hop lightpath (31, 28). After running the RWA algorithm, the one-hop lightpath may be routed over the path 28–30–32 (since node 28 is not directly connected to the hub node 32 of the virtual star), while the two-hop lightpath may in fact be routed over the direct link 31–28, completely bypassing the hub node 32 (unlike a physical star, where a two-hop lightpath is optically switched at the hub). Similar observations apply to all clusters at both levels of the hierarchy.

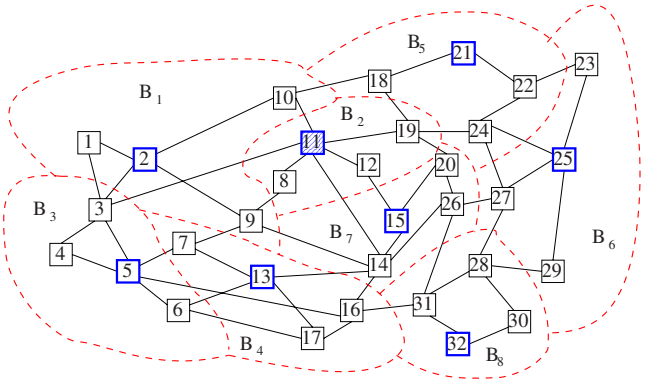

Fig. 1. (Color online) 32-node WDM network, partitioned into eight first-level clusters $B_1, \ldots, B_8$; hub nodes are blue squares and form a (logical) second-level cluster with hub 11.

III. Hierarchical Logical Topology Design

Suppose that the network has been partitioned into m clusters B_i with corresponding hub nodes h_i, $i = 1, \ldots, m$; we will describe such a clustering algorithm shortly. Our objective is to determine the logical topology and associated routing of the traffic demands $t^{(sd)}$ that minimizes the number of lightpaths; a secondary but important objective is to keep the number of required wavelengths low. The hierarchical logical topology algorithm we outline in this section reflects the two-level cluster hierarchy we described in the previous section and sets up lightpaths in a sequence of three steps, each step dealing with a different type of traffic demand: large demands, intracluster traffic, and intercluster traffic.

A. Setup of Direct Lightpaths for Large Traffic Demands

Some elements of the traffic demand matrix may be large enough to utilize the whole capacity of one or more wavelengths. For such demands $t^{(sd)}$, we simply assign as many direct lightpaths as necessary to carry them even if nodes s and d belong to different clusters. Given our goal of minimizing the total number of lightpaths in the logical topology, setting up direct lightpaths for such traffic is preferable to carrying it over a series of lightpaths with intermediate stops at hubs. We call this operation the *reduction* of the original matrix, after which all elements of the matrix will be smaller than the wavelength capacity C.

In this step, we also apply a "direct to the destination hub" rule to set up lightpaths between some source node s and a remote hub h, if the total amount of traffic from s to destination nodes d in h's cluster $\Sigma_d t^{(sd)} \geq p \times C$, where $p \in (0.5, 1)$ is a parameter determined by the network designer; in our work, we let $p = 0.8$. Setting up such lightpaths for large demands to bypass the local hub node (i.e., the hub in the cluster of node s) has several benefits: the number of lightpaths in the logical topology is reduced, the number of electronic ports and switching capacity required at hub nodes is reduced (leading to higher scalability), and the RWA algorithm may require fewer wavelengths (since hubs will be less of a bottleneck).

Let R_{init} be the set of direct lightpaths created in this step. Let $T_r = [t_r^{(sd)}]$ denote the matrix of residual traffic demands (i.e., excluding those carried by the lightpaths in R_{init}) that need to be groomed. Obviously, $t_r^{(sd)} < C$ for all s, d.

B. Intracluster Traffic Grooming

Consider the ith cluster B_i with n_i nodes and node h_i as its hub. We view cluster B_i as a virtual star with a $n_i \times n_i$ traffic matrix $T_i = [t_i^{(sd)}]$, defined as

$$t_i^{(sd)} = \begin{cases} t_r^{(sd)} & s \neq h_i, d \neq h_i \\ t_r^{(sd)} + \sum_{x \notin B_i} t_r^{(sx)} & d = h_i \\ t_r^{(sd)} + \sum_{x \notin B_i} t_r^{(xd)} & s = h_i \end{cases} . \quad (1)$$

In other words, if s and d are nonhub nodes, then $t_i^{(sd)}$ represents the intracluster traffic from s to d. If, on the other hand, node d (node s) is the hub node, then $t_i^{(sd)}$ includes not only the intracluster traffic component $t_r^{(sd)}$, but also the aggregate intercluster traffic originating at node s (terminating at node d). This definition of $t_i^{(sd)}$, when either s or d is the hub node, implements the hierarchical grooming of traffic: all intercluster traffic, other than that carried by direct lightpaths set up earlier, is first carried to the local hub, groomed there with intercluster traffic from other local nodes, carried on lightpaths to the destination hub (as we discuss shortly), groomed there with other local and nonlocal traffic, and finally carried to the destination node.

Given traffic matrix $T_i = [t_i^{(sd)}]$, we view cluster B_i as a virtual star with hub h_i and $n_i - 1$ nonhub nodes. We apply the StarTopology algorithm described in [14] to obtain the set of lightpaths R_i for carrying the demands $\{t_i^{(sd)}\}$. The lightpaths in R_i are either one hop (i.e., from a nonhub node to the hub, or vice versa), or two hop (i.e., from one nonhub node to another). Hence, the routing of the traffic components $t_i^{(sd)}$ is implicit in the logical topology R_i.

We emphasize that, at this stage, we only identify the lightpaths to be created; the routing of these lightpaths over the physical topology is performed during the third (RWA) phase of our approach. Depending on the actual topology of the cluster B_i, which may be quite different than that of a physical star, once routed, the lightpaths in R_i may follow paths that do not resemble at all the paths of a physical star. For instance, a one-hop lightpath from a nonhub node of the cluster to the hub h_i is routed on the unique link from the node to the hub in a physical star; in our case, however, the path followed by the lightpaths may consist of several links, depending on the physical topology of the network. Similarly, a two-hop lightpath is always switched optically at the hub of a physical star; in a virtual star cluster, on the other hand, a two-hop lightpath will be routed by the RWA algorithm on the actual underlying topology, and its path may not even pass through the hub h_i at all, if not doing so is more efficient in terms of resource usage (e.g., if the two nonhub nodes are connected by a direct link).

We perform intracluster grooming in this manner, by applying the StarTopology algorithm to each cluster $B_i, \ldots, B_m$, in isolation. As a result, at the end of this step, we identify a set of lightpaths $R_{\text{intra}} = R_1 \cup R_2 \cup \cdots \cup R_m$ for carrying all intracluster traffic.

C. Intercluster Traffic Grooming

At the end of intracluster grooming, all traffic (other than that carried by the initial direct lightpaths) from the nodes of a cluster B_i with destination outside the cluster, is carried to the hub h_i for grooming and transport to the destination hub. In order to groom this traffic, we consider a new cluster B that forms the second-level hierarchy in our approach. Cluster B consists of the m hub nodes $h_1, \ldots, h_m$, of the first-level clusters. Let $h \in \{h_1, \ldots, h_m\}$ be the node designated as the second-level hub. We view cluster B as a virtual star with an $m \times m$ traffic matrix $T_{\text{inter}} = [t_{\text{inter}}^{(h_i h_j)}]$ repre-

senting the intercluster traffic demands. This intercluster matrix is defined as

$$t_{\text{inter}}^{(h_i h_j)} = \sum_{s \in B_i, d \in B_j} t_r^{(s,d)}, \quad i,j = 1, \ldots, m, \quad i \neq j. \tag{2}$$

We now apply the StarTopology algorithm (see [14]) to the virtual star B with hub h, and we obtain the set of lightpaths R_{inter} to carry the traffic demands $\{t_{\text{inter}}^{(h_i h_j)}\}$. Again, we emphasize that the routing of these lightpaths is performed on the underlying physical topology; thus, the same observations regarding the routing of the intracluster lightpaths above also apply to the lightpaths in R_{inter}.

Figure 2 provides a pseudocode description of the hierarchical logical topology (MeshTopology) algorithm. The time complexity of the algorithm is determined by the application of the StarTopology algorithm for intracluster and intercluster grooming in Steps 5–8 and 13, respectively. Since the complexity of the StarTopology algorithm for a star with n nodes is $O(n^2)$, it is straightforward to see that the complexity of the MeshTopology algorithm for a network of N nodes is $O(N^2)$.

The outcome of the logical topology design phase is a set of lightpaths $R = R_{\text{init}} \cup R_{\text{intra}} \cup R_{\text{inter}}$ and an implicit routing of the original traffic components $t^{(sd)}$ over these lightpaths. This set R of lightpaths is the input to the RWA phase of our framework, which, as we discussed in the previous section, uses the LFAP algorithm [15] to route and color the lightpaths.

Hierarchical Logical Topology Algorithm for Mesh Networks
Input: A mesh WDM network with N nodes partitioned in m clusters $B_1, \ldots, B_m$, hub h_i of cluster B_i, W wavelengths per link, capacity C of each wavelength, and traffic matrix $T = [t^{(sd)}]$
Output: The set of lightpaths R in the logical topology, and the routing of the traffic components $t^{(sd)}$, such that $|R|$ is minimized

Procedure **MeshTopology**
begin // Set up direct lightpaths
1. Reduction: create direct lightpaths for demands $> C$
2. Direct to destination hub: create lightpaths to hub nodes when the aggregate traffic to a cluster is large ($\geq 0.8 \times C$)
3. $R_{init} \leftarrow$ initial set of direct lightpaths
// Intracluster grooming
4. $T_r = [t_r^{(sd)}] \leftarrow$ residual traffic matrix
5. **for** $i = 1, \ldots, m$ **do**
6. $T_i = [t_i^{(sd)}] \leftarrow$ intracluster traffic matrix for cluster B_i, computed from expression (1)
7. $R_i \leftarrow$ set of lightpaths obtained by running the StarTopology algorithm [14] on virtual star B_i w/ hub h_i
8. **end for**
9. $R_{intra} \leftarrow R_1 \cup R_2 \cup \cdots \cup R_m$
// Intercluster grooming
10. $B \leftarrow$ cluster consisting of m hub nodes $h_1, \ldots, h_m$
11. $h \leftarrow$ hub of cluster B
12. $T_{inter} \leftarrow$ the $m \times m$ intercluster matrix from expression (2)
13. $R_{inter} \leftarrow$ set of lightpaths obtained by running the StarTopology algorithm on virtual star B with hub h
14. Return the set of lightpaths $R = R_{init} \cup R_{intra} \cup R_{inter}$
end

Fig. 2. Hierarchical logical topology algorithm for mesh networks.

Finally, we note that we considered only two levels of clusters in our grooming algorithm. However, for networks of very large size, our approach can be extended to three or more levels of hierarchy in a straightforward manner.

IV. Clustering Algorithms in Network Design

Clustering is a function that arises frequently in problems related to network design and organization. A classic book [17] defines clustering as "grouping of similar objects" and discusses many mainstream clustering algorithms. The algorithms are classified as either *minimum cut* or *spanning tree*, depending on the underlying methodology. The input to the algorithms generally consist of a set of nodes (objects) and edge weights (node relationships), while the output is a partition of the nodes that optimizes a given objective function. In our case, the goal is to find a clustering that will minimize the number of lightpaths *after* applying the hierarchical grooming (logical design) approach, a fact that adds significant complexity to the problem. Specifically, the input to our problem consists of a traffic demand matrix and several constraints, in addition to the physical network topology; furthermore, unlike typical objective functions considered in the literature (e.g., the physical cut size or the amount of intercluster traffic), ours cannot be easily expressed as a function of the resulting clusters. Therefore, most of the existing clustering techniques are not directly applicable to the problem at hand.

Some clustering studies consider only the communication (traffic) pattern between nodes. For instance, an algorithm that can group a nearly completely decomposable matrix into blocks, so that the weighted arcs between blocks have values not exceeding a given threshold, was introduced in [18]. The algorithm, called TPABLO, can be used to group the states of large Markov chains. A similar objective exists in the traffic grooming context, as it is desirable for traffic demands within a cluster to be denser than intercluster traffic. However, the TPABLO algorithm does not take into account the physical topology; hence it may group together nodes that are far apart. Such clusters are inappropriate for the hierarchical logical topology we consider, since the long lightpaths created for intracluster traffic may significantly increase the number of wavelengths required in the whole network.

Other work has focused on the physical topology only. Typically, the goal is to partition the nodes into contiguous clusters containing roughly equal numbers of nodes, and at the same time minimize the overall cut size. An example is the work in [19] on multiobjec-

tive graph partitioning, which was implemented in the METIS software package. These algorithms were designed for VLSI (very-large-scale integration) design, a very different problem, where equality in size and a minimum of cross-layer connections are essential for each module, and are not applicable to traffic grooming. First, there is no requirement that all clusters be equal in size; more important, a small physical cut size may result in bottlenecks for intercluster traffic, which in turn may increase the wavelength requirements during the RWA phase.

Another family of clustering problems concerned with the physical network topology includes the well-known k-center, k-clustering, k-median, and facility location problems [20–23]. Unlike the applications targeted by METIS, they do not require clusters to be of equal size. Of all the variants, the k-center problem is of most interest to us. The goal of the k-center problem is to find a set S of K nodes (centers) in the network, so as to minimize the maximum distance from any network node to the nearest center. Thus, the set S implicitly defines K clusters with corresponding hub nodes in S.

A solution to the k-center problem may be useful for hierarchical traffic grooming, since it is likely to lead to short lightpaths within a cluster, thus lowering the wavelength requirements. Also, this type of clustering tends to avoid creating pathlike physical topologies for each cluster; pathlike topologies are not a good match for the StarTopology algorithm (described in [14]), which treats each cluster as a virtual star.

The k-center problem is NP-complete, and the best approximation ratio that can be obtained in polynomial time is 2 [24,25]. We implemented the 2-approximation algorithm in [24] for the k-center problem, and we compare it with our own clustering method in Section VII. For completeness, the steps of the algorithm are listed below:

1) Create a single cluster, $B_1=\{v_1,\ldots,v_n\}$, with hub node $h_1=v_1$. Calculate the all-pair shortest paths, record the distances in matrix *dist*, and let $x\leftarrow 1$.
2) Let x be the number of clusters and d be the maximum distance between any node and its hub, i.e., $d=\max\{dist(v_i,h_j)\},v_i\in B_j$. Let v be a node such that the distance between v and its hub is d.
3) Create a new cluster B_{x+1} with $h_{x+1}=v$ as the only node. Then for each node v', if v' is closer to v than to its current hub, move v' from its current cluster to the new cluster B_{x+1}. Let $x\leftarrow x+1$.
4) Repeat Steps 2 and 3 $K-1$ times, adding one cluster at each iteration, for a total of K clusters.

More recently, some studies have explored clustering techniques in the context of traffic grooming: a hierarchical design for interconnecting SONET rings with multirate wavelength channels was proposed in [26], and in [27], the blocking island paradigm is used to abstract network resources and find groups of bandwidth hierarchies for a restricted version of the traffic grooming problem. Our work, which we present in the next section, is more comprehensive, and it is applicable to many variants of the grooming problem.

V. Clustering for Hierarchical Grooming

We now describe a clustering algorithm tailored to the hierarchical grooming framework we propose. The objective of the algorithm is twofold: to partition the network into some number m of clusters, denoted $B_1,\ldots,B_m$, and to select one node in each cluster to serve as the hub where grooming of intracluster and intercluster traffic is performed. As we discussed in Section III, the clusters and hubs are the input to the subsequent logical topology design and RWA phases of the framework. Consequently, the quality of the clustering phase is an important factor in the quality of the overall design. Next, we discuss the trade-offs involved in selecting the clusters, which set the design principles for our clustering algorithm.

A. Important Considerations

To obtain a good clustering, the number of clusters, their composition, and the corresponding hubs must be selected in a way that helps achieve our goal of minimizing the number of lightpaths and wavelengths required to carry the traffic demands. Therefore, the selection of clusters and hubs is a complex and difficult task, as it depends on both the physical topology of the network and the traffic matrix T. To illustrate this point, consider the trade-offs involved in determining the number m of clusters. If m is small, the amount of intercluster traffic will likely be large. Hence, the m hubs may become bottlenecks, resulting in a large number of electronic ports at each hub and possibly a large number of wavelengths (since many lightpaths may have to be carried over the fixed number of links to and from each hub).

On the other hand, a large value for m implies a small number of nodes within each cluster. In this case, the amount of intracluster traffic will be small, resulting in inefficient grooming (i.e., a large number of lightpaths); similarly, at the second-level cluster, $O(m^2)$ lightpaths will have to be set up to carry small amounts of intercluster traffic. Therefore, the network designer must select the number and size of clusters to strike a balance between capacity utilization and

number of lightpaths for both intracluster and intercluster traffic.

Now consider the composition of each cluster. If the average traffic demand between nodes within a cluster is higher than the average intercluster demand, there will tend to be fewer intercluster lightpaths, which are typically longer than local lightpaths. Therefore, it is desirable to cluster together nodes with denser traffic between each other: doing so reduces the number of longer lightpaths, alleviates hub congestion, and provides more flexibility to the RWA algorithm (since long lightpaths are more likely to collide during the routing and wavelength assignment phase).

On the physical topology side, we also need to consider the cut links that connect different clusters. Each cluster has a number of fibers that link to nodes outside the cluster, and all traffic between a node outside the cluster and one within must traverse these cut links. Since the cut links must have sufficient capacity to carry the intercluster traffic, it is important to select clusters such that their cut size is not too small, in order to keep the wavelength requirements low.

Another important consideration arises in physical topologies for which there exists a critical small cut set that partitions the network into two parts. In such a topology, all traffic between the two sides of the bisection will have to go through the cut. In this case, creating clusters that consist of nodes on different sides of the cut may be undesirable, because it may generate unnecessary traffic that goes back and forth through the cut. Consider a cluster with nodes i,j, on one side of the bisection and the hub h on the other. Due to the nature of the hierarchical grooming approach, traffic between i and j may need to be sent to the hub first, creating additional traffic across the cut links, with a corresponding increase in the number of required wavelengths. This additional traffic can be eliminated by forcing nodes on different sides of the bisection to be in different clusters. We describe shortly a precutting technique that can be useful in such situations.

The physical shape of each cluster may also affect the wavelength requirements. In particular, it is important to avoid the creation of clusters whose topology resembles that of a path, since in such topologies the links near the hub can become congested. Since we use a virtual star approach for logical topology design within each cluster, topologies with relatively short diameter are more attractive in terms of RWA.

In the next subsection, we describe an algorithm that takes into account all the above factors in partitioning the network into clusters to yield good grooming solutions.

B. MeshClustering Algorithm

Figure 3 provides a pseudocode description of our MeshClustering algorithm, which we use to partition a network of general topology in order to apply our hierarchical traffic grooming framework. The algorithm includes several user-defined parameters that can be used to control the size and composition of clusters, either directly or indirectly. Parameters *MinCS* and *MaxCS* represent the minimum and maximum cluster size, respectively. Our algorithm treats these parameters as an indication of the desirable range of cluster sizes, rather than as hard thresholds that cannot be violated. Although the algorithm attempts to keep the size of each cluster between the values of these two parameters (inclusive), it has the freedom, based on the values of the other parameters, to determine what it thinks may be the best clustering. Consequently, the final result may contain clusters larger than

A Clustering Algorithm for Mesh Networks
Input: A mesh network with a set V of $|V| = N$ nodes, capacity C for each wavelength, and reduced traffic matrix $T_r = [t_r^{(sd)}]$.
User-defined parameters: $MinCS, MaxCS$ for the desired minimum and maximum cluster size, respectively, a threshold $0.5 \leq \Delta \leq 0.8$, a cluster diameter-to-nodes ratio $0 < \delta \leq 0.75$, and an intra- to intercluster traffic ratio $0.8 \leq \rho \leq 1.25$.
Output: A partitioning of the node set V into some number m of clusters, $B_1, \ldots, B_m$, and the selection of node h_i as the hub of cluster B_i, such that the size of each cluster is roughly between $MinCS$ and $MaxCS$ and the clustering will lower the lightpath and wavelength requirements of the subsequent hierarchical logical topology design and RWA algorithms.

Procedure **MeshClustering**
begin
while $V \neq \phi$ **do**
  $v \leftarrow$ node in V with maximum remaining capacity
  $B \leftarrow \{v\}$ // new cluster B with hub v
  $V \leftarrow V - \{v\}$
  while $V \neq \phi$ and $|B| < MaxCS$ **do** // grow cluster B
    $Q \leftarrow$ set of nodes $\in V$ adjacent to nodes in B
    foreach node $q \in Q$ **do**
      $B' \leftarrow B \cup \{q\}$ // assume q is included in B
      HUBTEST: does traffic between B', $\overline{B'}$ occupy more than Δ of the remaining hub capacity?
      CUTTEST: does traffic between B', $\overline{B'}$ occupy more than Δ of the remaining cut link capacity?
      if q passes both tests **then**
        $x \leftarrow$ total traffic between q and nodes in B
        $y \leftarrow$ total traffic between q and nodes in $\overline{B'}$
        $\rho_q \leftarrow x/y$ //intra- to intercluster traffic ratio
        $d \leftarrow$ diameter of induced subgraph B'
        $\delta_q \leftarrow d/|B'|$ // diameter-to-nodes ratio
      else $Q \leftarrow Q - \{q\}$
    end for
    if $Q = \phi$ **then break** // cannot grow cluster B
    else
      $q_0 \leftarrow$ node $\in Q$ with largest ρ_q and smallest δ_q
      $B \leftarrow B \cup \{q_0\}$ // grow cluster B to include q_0
      $V \leftarrow V - \{q_0\}$
  end while // continue until B cannot grow further
end while
Combine clusters of size $< MinCS$ with adjacent clusters
end

Fig. 3. Clustering algorithm for mesh networks.

MaxCS (see also the discussion below regarding Step 26 of the algorithm).

The parameter Δ ($0.5 \leq \Delta \leq 0.8$, default value $\Delta = 0.8$) is used to test whether there is sufficient capacity at the hub node, as well as the edges connecting the cluster to the rest of the network, to groom and carry the traffic demands. Specifically, we require that the intercluster traffic originating from or terminating at a given cluster not exceed a fraction Δ of the hub capacity (this is the HUBTEST in Step 9 of the algorithm); similarly, this intracluster traffic must not exceed a fraction Δ of the capacity of the links connecting the cluster to the rest of the network (the CUTTEST in Step 10 of the algorithm). The algorithm will consider a node to add to a cluster only if doing so will not violate these two constraints.

The parameter δ is meant to control the ratio of the diameter of a cluster to the number of nodes it contains. In order to avoid cluster topologies that resemble long paths, we require that $0 < \delta \leq 0.75$. We used the value $\delta = 0.75$ in our experiments; this value corresponds to a four-node path, hence restricting the longest path within a cluster to no more than three links. Finally, the parameter ρ, $0.8 \leq \rho \leq 1.25$, specifies the acceptable range for the ratio of intracluster-to-intercluster traffic for a given cluster. As we discussed earlier, it is desirable to cluster together nodes that exchange a substantial amount of traffic among themselves relative to traffic they exchange with the rest of the network; therefore, we let $\rho = 1.25$ in our implementation.

The MeshClustering algorithm in Fig. 3 generates one cluster during each iteration of the main **while** loop between Steps 1 and 25. Initially, in Steps 2–4, the hub of a new cluster B is selected as the node with the maximum remaining capacity among those not yet assigned to a cluster; by "remaining capacity," we mean the capacity remaining on its incident links after subtracting the bandwidth taken up by any direct lightpaths set up as discussed in Section III.A. Following the hub selection, we grow the cluster by adding one node during each iteration of the innermost **while** loop between Steps 5 and 24. At each iteration, the set Q of candidate nodes for inclusion in cluster B consists of all nodes, not yet assigned to another cluster, that are adjacent to nodes in B. For each node $q \in Q$, we first check whether including q in B would result in a cluster that passes both the HUBTEST (Step 9) and CUTTEST (Step 10); if not, node q is removed for consideration for inclusion into cluster B (Step 17). For all nodes q that pass both tests, we compute the diameter-to-nodes ratio δ_q and intracluster-to-intercluster traffic ratio ρ_q, assuming that q is added to cluster B (Steps 11–16). Let q_0 be a node that passes both tests and has the largest ρ_q value among the candidates; if there are multiple such nodes, we select the one with the smallest δ_q value. We include q_0 in cluster B (Steps 21–23), and the process is repeated as long as the size of B is less than *MaxCS*.

Once all nodes have been assigned to clusters, it is possible for one or more of the clusters to have fewer than *MinCS* nodes. In this case, at Step 26, the algorithm removes these clusters and includes their nodes into adjacent clusters. As a result, at the end of the algorithm some clusters may contain more than *MaxCS* nodes.

The running time complexity of the algorithm is determined by Steps 1–25. The main **while** loop executes a number of times equal to the number m of clusters; for each cluster, the innermost **while** loop executes a number of times equal to the size of the cluster. Therefore, Steps 5–24 of the algorithm are repeated N times, where N is the number of nodes in the network. Each time, the **foreach** loop executes at most N times. Steps 9, 10, 12, and 13 each take no more than $O(N^2)$ time in the worst case, while all other steps take constant time. Therefore, the asymptotic complexity of the algorithm is $O(N^4)$. However, this bound is quite loose; in practice, we have found that the algorithm takes only a few seconds for the 128-node, 321-link network we consider in Section VII.

C. Precutting for Imbalanced Topologies

As we mentioned in Subsection V.A, when the topology has a bisection of small cut size, the cut links are likely to become congested, as they have to carry all traffic between the two parts of the network on either side of the bisection. To reduce congestion, and hence the number of required wavelengths, in such topologies, it may be necessary to disallow nodes on different sides of the bisection from being in the same cluster. However, identifying such a critical bisection in a large, imbalanced topology, is a difficult task. In Section VI.B, we describe a method we developed for this purpose and which we also use to obtain a lower bound on the number of wavelengths.

Once we identify a critical bisection, we apply the following approach. First, we use the MeshClustering algorithm to determine a clustering that does not take the bisection into consideration. Then, we partition the network into two parts along the bisection, and we apply the MeshClustering algorithm on each part separately; this ensures that no cluster contains nodes from both sides of the bisection. We then select the clustering that requires the fewest lightpaths after the logical topology and RWA phases, unless it requires a significantly larger number (e.g., 10% or more) of wavelengths; in this way, we achieve balance between the lightpath objective and the wavelength requirements.

VI. Lower Bounds

We now obtain lower bounds on both the number of lightpaths and the number of wavelengths required to carry the traffic matrix T. These bounds are obtained independently of the manner (e.g., hierarchical or otherwise) in which traffic grooming is performed; hence they are useful in characterizing the effectiveness of our algorithm.

A. Integer Linear Programming Lower Bound on Number of Lightpaths

A simple lower bound F^l on the total number of lightpaths (our main objective) can be obtained as

$$F^l = \max\left(\sum_s \left\lceil \frac{\sum_d t^{(sd)}}{C} \right\rceil, \sum_d \left\lceil \frac{\sum_s t^{(sd)}}{C} \right\rceil\right). \tag{3}$$

This bound is based on the observation that each node must source and terminate a sufficient number of lightpaths to carry the traffic demands from and to this node, respectively. This bound can be determined directly from the traffic matrix T.

It is possible to obtain a better lower bound by using a standard ILP relaxation technique. Starting from the ILP formulation (e.g., see [3]) of the traffic grooming problem, we remove some variables and constraints, so that we can obtain a relaxed solution that serves as a lower bound for the original objective. In the relaxed ILP formulation, let b_{sd} denote the number of direct lightpaths set up from s to d. Since all traffic originating at source node s must be carried on some lightpath also originating at s, the following constraints must be observed:

$$\sum_d b_{sd} C \geq \sum_d t^{(sd)} \quad \forall\, s. \tag{4}$$

Similarly, for each destination d we have that

$$\sum_s b_{sd} C \geq \sum_s t^{(sd)} \quad \forall\, d. \tag{5}$$

Since our goal is to minimize the overall number of lightpaths, the resulting relaxed ILP formulation is

Minimize: $\Sigma_{s,d} b_{sd}$

Subject to: Constraints (4) and (5).

We emphasize that this ILP will not necessarily yield a meaningful solution to the original grooming problem, only a lower bound. By configuring CPLEX to use dual steepest-edge pricing, we are able to compute this bound within a few seconds even for the 128-node topology that we consider in the next section. Although this bound is better than the simple bound F^l above, we believe that it is somewhat loose. However, we have found that introducing additional constraints on traffic flow and/or routing into the relaxed ILP in order to improve the lower bound tends to increase the running time of CPLEX substantially, to the point that it becomes impractical for the large networks we consider in this work. Therefore, we use the lower bound obtained from the above simple ILP in our experimental study.

B. Lower Bound on Number of Wavelengths

Consider a bisection cut of the network, and let t be the maximum amount of traffic that needs to be carried on either direction of the links in the cut set. If x is the number of links in the cut set, then the quantity $\lceil t/xC \rceil$ is a lower bound on the number of wavelengths for carrying the given traffic matrix. Computing this bound does not require any information regarding the logical topology or the routing and wavelength assignment of lightpaths.

The main challenge, then, is to find a bisection of the network that yields a good lower bound. To this end, we make the observation that there is a trade-off between the cut size and the relative sizes of the node sets at each side of the bisection. On the one hand, using software such as METIS [19] to divide the network into two equal parts may not yield a good bound if, due to the irregular nature of the topology, the cut size turns out to be large. On the other hand, a small cut size may not be effective either, if one node set is significantly larger than the other, resulting in little traffic across the cut links.

To reconcile these conflicting objectives, we used the CHACO software [28], which implements the partitioning algorithm in [29]. The software uses the parameter KL-IMBALANCE to control the relative sizes of the node sets on either size of the bisection. To determine a cut that achieves a good balance between the two objectives, we apply CHACO several times, varying the KL-IMBALANCE parameter, and obtain several different bisections of the physical topology. We then select the bisection that corresponds to the best (highest) lower bound, under the assumption of uniform traffic. We use this bisection to calculate wavelength bounds for all problem instances on this topology, even though the actual traffic matrix is generated based on a different pattern (discussed shortly).

VII. Numerical Results

In this section, we present experimental results to demonstrate the performance of our clustering and hierarchical grooming algorithms. The traffic matrix $T = [t^{(sd)}]$ of each problem instance we consider is generated by drawing $N(N-1)$ random numbers (rounded

to the nearest integer) from a Gaussian distribution with a given mean t and standard deviation σ that depend on the traffic pattern. We consider three traffic patterns here:

1) *Random pattern.* Random patterns are often challenging, since the matrix does not have any particular structure that can be exploited by a grooming algorithm. To generate a traffic matrix, we let the standard deviation σ be 150% of the mean t. Consequently, the traffic elements $t^{(sd)}$ take values in a wide range around the mean, and the link loads also vary widely. If the random number generator returns a negative value for some element, we set the corresponding $t^{(sd)}$ value to zero.
2) *Falling pattern.* This pattern is designed to capture the locality property that has been observed in some networks. Specifically, if the mean of the distribution for node pairs that have shortest distance 1 is t, then the mean for node pairs with shortest distance 2 (3) is set to $0.8t$ ($0.6t$); for all other pairs, the mean is set to $0.2t$. We also let the standard deviation σ be 20% of the mean.
3) *Rising pattern.* This is the opposite of the falling pattern. We use t as the mean for traffic demands between node pairs with the longest path. Node pairs with the second and third longest paths have mean values $0.8t$ and $0.6t$, respectively. All other node pairs have mean $0.2t$. The standard deviation is also set to 20% of the mean.

For a given network topology and traffic pattern, we generate 30 problem instances, and we compare our MeshClustering algorithm (shown in Fig. 3) to the k-center algorithm [24] we described in Section IV. We consider two performance metrics in our study: the *normalized lightpath count* and the *normalized wavelength count*. The former is the ratio of the number of lightpaths required for hierarchical traffic grooming, when clustering is performed by one of the two algorithms above, to the lightpath lower bound obtained using the relaxed ILP in Subsection VI.A; the latter is the ratio of the number of wavelengths required to the wavelength lower bound computed as explained in Subsection VI.B. The normalized metrics filter out the effect of the traffic matrix, allowing us to compare results among problem instances created by very different traffic patterns; obviously, a smaller value of the two metrics implies a better solution. We also consider two network topologies, one in each of the following two subsections. We emphasize that the size of the second topology is about an order of magnitude larger than the typical topology considered in previous grooming studies, a fact that demonstrates the scalability of our hierarchical grooming approach.

A. 47-Node Balanced Topology Network

We first consider a 47-node, 96-link network topology that appeared in a historical paper on network design [30]. The node degree of the network is relatively high, and the topology is balanced, in the sense that there is no bisection with a small cut size that can be a bottleneck in traffic grooming.

Figures 4 and 5 plot the normalized lightpath and wavelength count, respectively, for each of 30 problem instances whose traffic matrix was generated according to the falling pattern. For each problem instance, four values are shown, corresponding to four different clusterings. The first two are from the k-center algorithm, with the number of clusters K equal to 4 and 6, respectively. The other two are from our MeshClustering algorithm. Recall that our algorithm does not take the number of clusters as input; rather, it tries to optimize it. Consequently, the algorithm may produce different clusters for two different problem instances, even if they are defined on the same topology and their matrices are drawn from the same distribution. To make the comparison against the k-center algorithm as fair as possible, we selected two sets of values for the user-defined parameters of MeshClustering (refer to Fig. 3) so that the average number of clusters over all 30 instances are 3.52 and 5.45, respectively.

Let us first consider the normalized lightpath count. From Fig. 4, we observe that the number of lightpaths required for hierarchical grooming is about 40% higher than the lower bound, regardless of the clustering algorithm used. Recall that the lower bounds were obtained by relaxing most of the constraints in the ILP formulation; hence we believe that they are rather loose. Therefore, these results (as well as the ones to be discussed shortly) serve as a validation of our approach, as they demonstrate that the lightpath requirements for hierarchical grooming are close to optimal. We also observe that, except for a couple of instances, the curves corresponding to the MeshClustering algorithm lie below those corresponding to the k-center algorithm. Although the difference is not high, we would like to point out that a 1% reduction in the number of lightpaths in this network with relatively dense demands would result in about 40 fewer electronic ports, a substantial savings in cost.

Let us now turn our attention to the normalized wavelength count. As Fig. 5 demonstrates, the MeshClustering algorithm requires significantly fewer wavelengths than the k-center algorithm. This result is due to the fact that unlike the k-center algorithm, ours is designed to take the wavelength requirements into account. In absolute terms, the difference in the number of wavelengths for these problem instances is of the order of 10–12. Put another way, given a con-

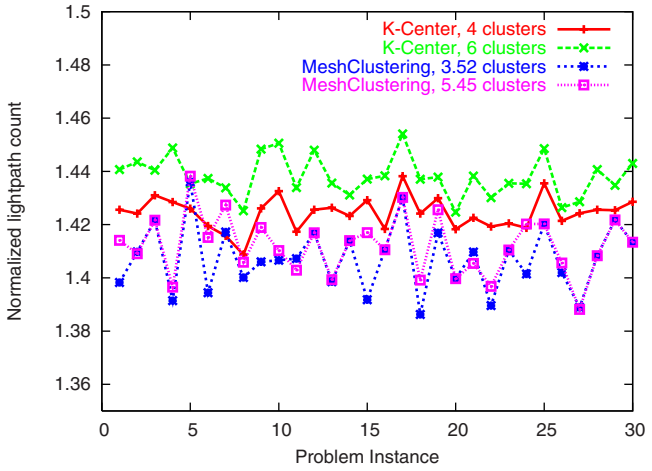

Fig. 4. (Color online) Lightpath comparison, falling pattern, 47-node network.

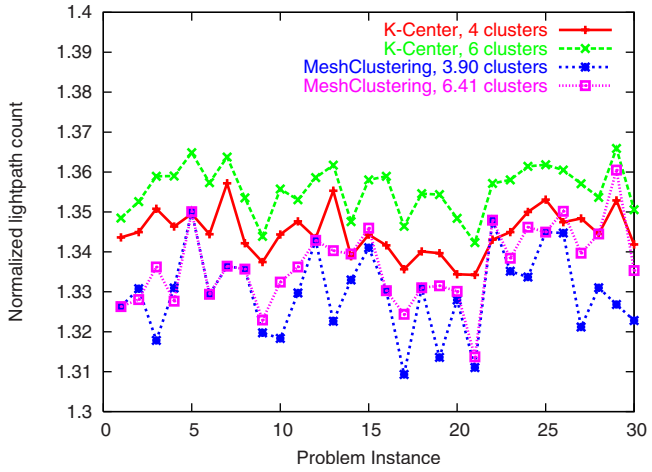

Fig. 6. (Color online) Lightpath comparison, rising pattern, 47-node network.

straint on the number of available wavelengths, our algorithm is more likely to generate clusters that admit a feasible grooming solution. We also note that the large values of the normalized wavelength count are due to the fact that the lower bound is loose in this case. Recall that a good bound depends on finding a network cut of small size and large crosscut traffic. However, this particular 47-node network was designed to avoid such a bottleneck cut. Furthermore, the falling traffic pattern makes it unlikely that a large amount of traffic will cross any network cut; as we shall see in a moment, the rising pattern yields better bounds.

Figures 6 and 7 are similar to Figs. 4 and 5, respectively, but show results for the rising traffic pattern. Note that, due to the nature of this pattern, relatively large amounts of traffic will cross any network cut, resulting in the much tighter wavelength bounds in Fig. 7. Again, except for a few instances, our clustering algorithm outperforms the k-center algorithm. Regardless of the clustering algorithm, we observe that hierarchical grooming is also close to optimal, confirming our earlier observations.

We have obtained similar results for the random and other traffic patterns, which we omit because of space constraints; these results can be found in [14].

B. 128-Node Imbalanced Topology Network

We now consider a 128-node, 321-link network, which corresponds to the worldwide backbone operated by a large service provider; we obtained the topology information from data documented on CAIDA's web site (http://www.caida.org). This topology is imbalanced, in the sense that there exists a bisection with a small cut size of 5 links that divides it into two parts of 114 and 14 nodes, respectively. We identify this critical cut with the method discussed in Subsection VI.B and use it to calculate the lower bound on the number of wavelengths. We also steer our algo-

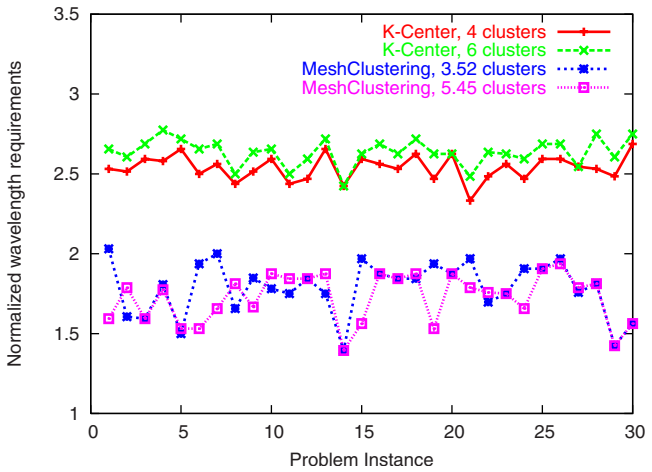

Fig. 5. (Color online) Wavelength comparison, falling pattern, 47-node network.

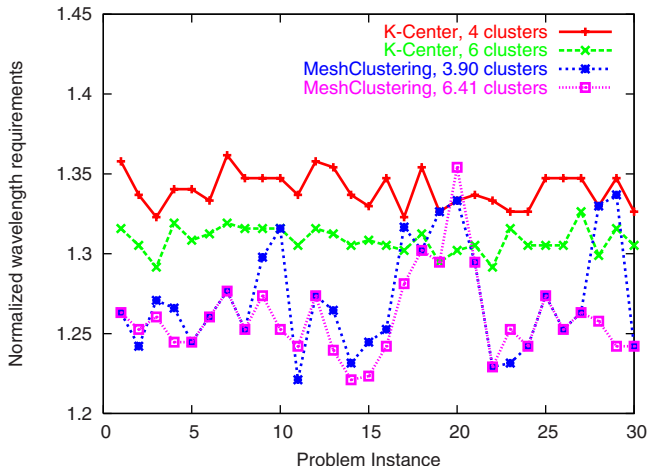

Fig. 7. (Color online) Wavelength comparison, rising pattern, 47-node network.

rithm towards creating clusters that contain nodes on one side of the cut only, as we explained in Subsection V.C.

Figures 8 and 9 plot the normalized lightpath and wavelength count, respectively, for 30 instances generated according to the random traffic pattern; Figs. 10 and 11 show similar plots but for instances generated according to the rising pattern. Additional results for other traffic patterns may be found in [14]. For the k-center algorithm, we let the number K of clusters be either 9 or 10, and we selected the parameters of the MeshClustering algorithm so that it also produces either 9 or 10 clusters (the average over all instances is 9.33 and 9.03 for the random and rising patterns, respectively). As we can see, our clustering algorithm slightly outperforms k-center in terms of the number of lightpaths, and both algorithms are relatively close to the (loose) lower bound. However, in terms of the number of wavelengths, our algorithm produces results that are within 5% of the lower bound, whereas k-center requires more than twice the number of wavelengths of our algorithm.

VIII. Concluding Remarks

Hierarchical traffic grooming is a new approach for efficient and cost-effective design of large-scale optical networks with multigranular traffic demands. We have demonstrated that the hierarchical grooming framework must be coupled with clustering techniques that follow grooming-specific design principles; we have presented such a clustering algorithm that is flexible in balancing various conflicting goals via user-defined parameters. We have also developed practical methods for computing lower bounds on metrics of interest. We have applied our algorithms to networks of realistic size. Overall, our experimental results demonstrate that (1) our hierarchical grooming approach scales to very large networks; (2) our clustering algorithm outperforms algorithms that were not developed with traffic grooming in mind; and (3) hierarchi-

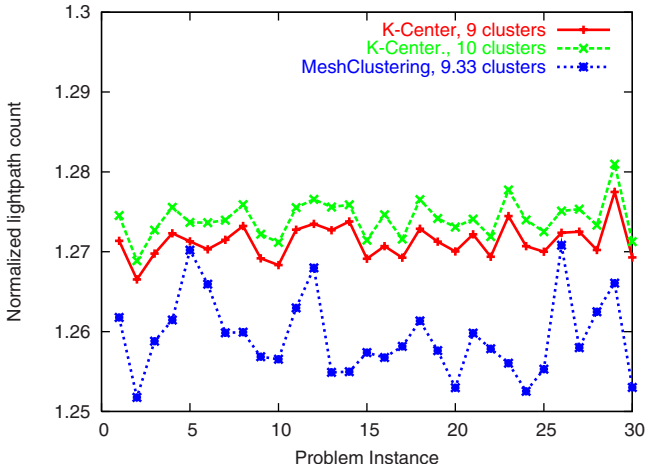

Fig. 8. (Color online) Lightpath comparison, random pattern, 128-node network.

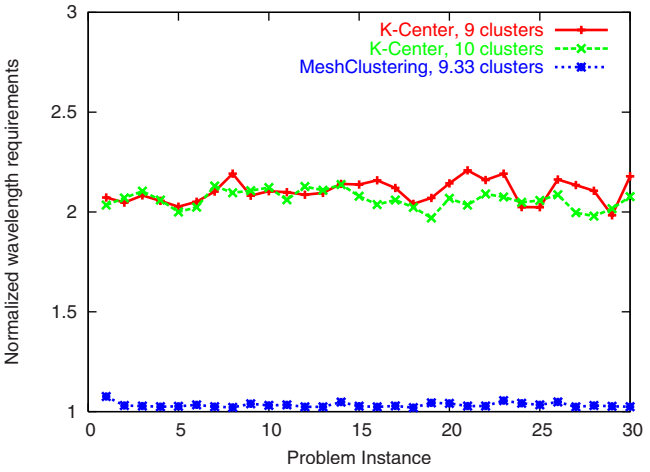

Fig. 9. (Color online) Wavelength comparison, random pattern, 128-node network.

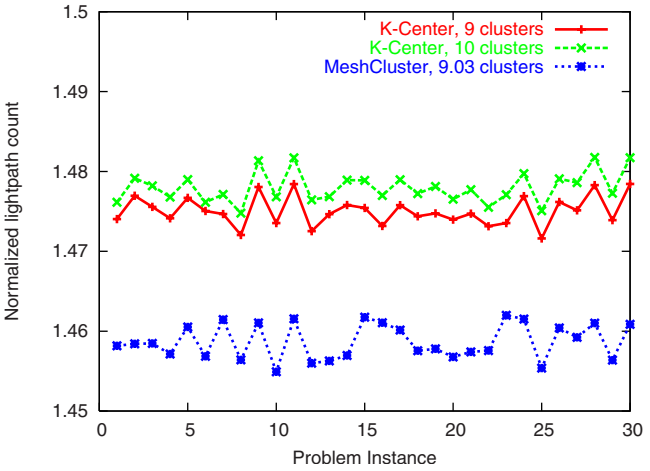

Fig. 10. (Color online) Lightpath comparison, rising pattern, 128-node network.

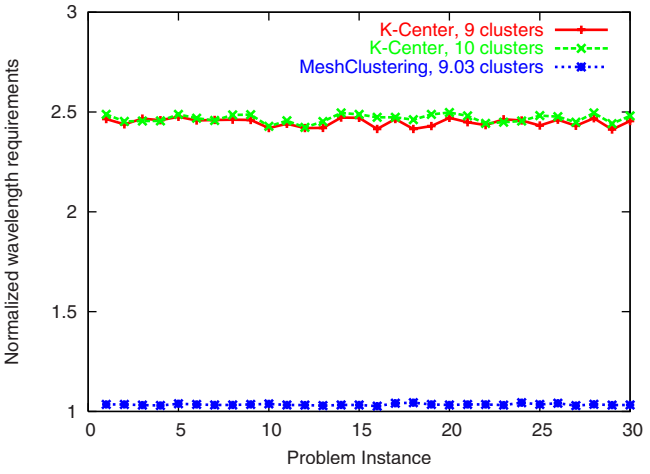

Fig. 11. (Color online) Wavelength comparison, rising pattern, 128-node network.

cal grooming combined with specially designed clustering techniques produce logical topologies that perform well in terms of both lightpath and wavelength requirements across a variety of traffic patterns.

ACKNOWLEDGMENT

This work was supported by National Science Foundation (NSF) grant ANI-0322107.

REFERENCES

[1] R. Dutta and G. N. Rouskas, "On optimal traffic grooming in WDM rings," *IEEE J. Sel. Areas Commun.*, vol. 20, no. 1, pp. 110–121, Jan. 2002.

[2] H. Zhu, H. Zang, K. Zhu, and B. Mukherjee, "A novel generic graph model for traffic grooming in heterogeneous WDM mesh networks," *IEEE/ACM Trans. Netw.*, vol. 11, no. 2, pp. 285–299, Apr. 2003.

[3] R. Dutta and G. N. Rouskas, "Traffic grooming in WDM networks: past and future," *IEEE Network*, vol. 16, no. 6, pp. 46–56, Nov./Dec. 2002.

[4] O. Gerstel, R. Ramaswami, and G. Sasaki, "Cost-effective traffic grooming in WDM rings," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 618–630, Oct. 2000.

[5] P.-J. Wan, G. Calinescu, L. Liu, and O. Frieder, "Grooming of arbitrary traffic in SONET/WDM BLSRs," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 10, pp. 1995–2003, 2000.

[6] V. R. Konda and T. Y. Chow, "Algorithm for traffic grooming in optical networks to minimize the number of transceivers," in *IEEE Workshop on High Performance Switching and Routing*, Dallas, TX, 2001, pp. 218–221.

[7] D. Li, Z. Sun, X. Jia, and K. Makki, "Traffic grooming on general topology WDM networks," *IEE Proc.–Commun.*, vol. 150, no. 3, pp. 197–201, June 2003.

[8] J. Hu and B. Leida, "Traffic grooming, routing, and wavelength assignment in optical WDM mesh networks," *Proc. of the IEEE INFOCOM 2004, the 23rd Annu. Joint Conf. of the IEEE Computer and Communications Societies*, Hong Kong, 2004, vol. 1, pp. 495–501.

[9] Z. K. G. Patrocínio, Jr., and G. R. Mateus, "A Lagrangian-based heuristic for traffic grooming in WDM optical networks," in *IEEE Global Telecommunications Conf., 2003. GLOBECOM '03.*, 2003, vol. 5, pp. 2767–2771.

[10] C. Lee and E. K. Park, "A genetic algorithm for traffic grooming in all-optical mesh networks," in *2002 IEEE Int. Conf. on Systems, Man and Cybernetics*, 2002, vol. 7.

[11] B. Chen, G. N. Rouskas, and R. Dutta, "On hierarchical traffic grooming in WDM networks," *IEEE/ACM Trans. Netw.*, vol. 16, no. 5, pp. 1226–1238, Oct. 2008.

[12] I. Chlamtac, A. Ganz, and G. Karmi, "Lightpath communications: an approach to high bandwidth optical WANS," *IEEE Trans. Commun.*, vol. 40, no. 7, pp. 1171–1182, July 1992.

[13] B. Chen, G. N. Rouskas, and R. Dutta, "Traffic grooming in WDM ring networks to minimize the maximum electronic port cost," *Opt. Switching Networking*, vol. 2, no. 1, pp. 1–18, May 2005.

[14] B. Chen, "Hierarchical traffic grooming in large-scale WDM networks," Ph.D. dissertation, North Carolina State University, Raleigh, NC, August 2005.

[15] H. Siregar, H. Takagi, and Y. Zhang, "Efficient routing and wavelength assignment in wavelength-routed optical networks," *Proc. 7th Asia-Pacific Network Operations and Management Symp.*, Fukuoka, Japan, 2003, pp. 116–127.

[16] S. Baroni and P. Bayvel, "Wavelength requirements in arbitrary connected wavelength-routed optical networks," *J. Lightwave Technol*, vol. 15, no. 2, pp. 242–251, Feb. 1997.

[17] J. A. Hartigan, *Clustering Algorithms*, New York: Wiley, 1975.

[18] H. Choi and D. B. Szyld, "Application of threshold partitioning of sparse matrices to Markov chains," in *Proc. of the IEEE Int. Computer Performance and Dependability Symp.*, 1996, pp. 158–165.

[19] K. Schloegel, G. Karypis, and V. Kumar, "A new algorithm for multi-objective graph partitioning," Department of Computer Science, University of Minnesota, Tech. Rep. 99–003, Sept. 1999.

[20] J. Bar-Ilan, G. Kortsarz, and D. Peleg, "How to allocate network centers," *J. Algorithms*, vol. 15, pp. 385–415, 1993.

[21] D. Hochbaum and D. Shmoys, "A unified approach to approximation algorithms for bottleneck problems," *J. ACM*, vol. 33, pp. 533–550, 1986.

[22] D. B. Shmoys, "Approximation algorithms for facility location problems," in *Approximation Algorithms for Combinatorial Optimization: Proc. Third Int. Workshop, APPROX 2000*, Saarbrücken, Germany, 2000, vol. 1913, pp. 27–32.

[23] M. Thorup, "Quick k-median, k-center, and facility location for sparse graphs," *Automata, Languages and Programming: Proc. of the 28th Int. Colloquium, ICALP 2001*, Crete, Greece, 2001, vol. 2076, pp. 249–260.

[24] T. Gonzalez, "Clustering to minimize the maximum intercluster distance," *Theoret. Comput. Sci.*, vol. 38, pp. 293–306, 1985.

[25] D. Hochbaum and D. Shmoys, "A best possible heuristic for the k-center problem," *Math. Op. Res.*, vol. 10, pp. 180–184, 1985.

[26] M. Esfandiari, S. Gloeckle, A. Zolfagheri, G. Clapp, J. Gannett, H. Kobrinski, V. Poudyal, and R. Skoog, "Improved metro network design by grooming and aggregation of STS-1 demands into OC-192/OC-48 lightpaths," in *The Nat. Fiber Optic Engineers Conf. (NFOEC) 2003*, Orlando, FL, 2003.

[27] Z. Ding and M. Hamdi, "Clustering techniques for traffic grooming in optical WDM mesh networks," in *2002 IEEE Global Telecommunications Conf., GLOBECOM 2002*, vol. 3, 2002, pp. 2711–2715.

[28] B. Hendrickson and R. Leland, "The CHACO user's guide," Sandia National Laboratories, Albuquerque, NM, Tech. Rep. SAND95–2344, July 1995.

[29] B. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," *Bell Syst. Tech. J.*, vol. 29, pp. 291–307, 1970.

[30] P. Baran, "On distributed communications networks," *IEEE Trans. Commun.*, vol. 12, no. 1, pp. 1–9, Mar. 1964.