Clustering for Hierarchical Traffic Grooming in Large Scale Mesh WDM Networks *

Bensong Chen[1], Rudra Dutta[2], and George N. Rouskas[2]

[1] Google Labs
[2] North Carolina State University

Abstract. We present a clustering algorithm for hierarchical traffic grooming in large WDM networks. In hierarchical grooming, the network is decomposed into clusters, and one hub node in each cluster is responsible for grooming traffic from and to the cluster. Hierarchical grooming scales to 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 tradeoff among various goals.

1 Introduction

Traffic grooming, the area of research concerned with efficient and cost-effective transport of sub-wavelength traffic over multigranular networks, has emerged as an important field of study in optical networks. In *static* grooming [5], the objective is to provision the network to carry a set of long-term traffic demands while minimizing the overall network cost. In *dynamic* grooming [18], on the other hand, the goal is to develop on-line algorithms for efficiently grooming and routing of connections that arrive in real time. The reader is referred to [6] for a comprehensive survey and classification of research on traffic grooming.

Most grooming studies on general topologies [12, 15] 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, since the running-time complexity of traffic grooming algorithms increases rapidly with the size of the network and also the operation, management, and control of multigranular networks becomes a challenging issue in large, unstructured topologies. Indeed, in existing networks, resources are typically managed and controlled in a hierarchical manner.

Based on this observation, we have developed a scalable hierarchical approach to traffic grooming [2] modeled after the hub-and-spoke paradigm used within the airline industry. The network is partitioned into clusters, and one node within each cluster serves as the *hub*. Non-hub nodes route their traffic to the hub, where it is groomed and forwarded to the destination cluster; hence, hubs are the only nodes responsible for grooming traffic not originating/terminating locally.

* This work was supported by the NSF under grant ANI-0322107.

I. Tomkos et al. (Eds.): ONDM 2007, LNCS 4534, pp. 249–258, 2007.

In this work we address an important yet challenging issue in hierarchical grooming, namely, 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 develop a new parameterized clustering algorithm that is flexible and allows the designer to achieve a balance among a number of conflicting goals. To demonstrate the effectiveness of hierarchical grooming with clustering, we present results for two large networks, including a 128-node, 321-link topology that is approximately an order of magnitude larger than networks that have been considered in previous grooming studies.

We describe the hierarchical grooming approach in Section 2. In Section 3 we present the clustering algorithm, and we obtain lower bounds in Section 4. We present numerical results in Section 5 and we conclude the paper in Section 6.

2 Hierarchical Grooming in Mesh Networks

We consider a network of general topology with N nodes. Each link supports W wavelengths, and the wavelength capacity C is an integer multiple of a basic transmission unit. Traffic demands are provided in matrix $T = [t^{(sd)}]$, where integer $t^{(sd)}$ denotes the amount of traffic to be carried from node s to node d.

The objective of the traffic grooming problem is to determine the lightpaths to be set up so as to carry the traffic matrix T while minimizing the number of electronic ports required. Since each lightpath requires exactly two electronic ports, this objective is equivalent to minimizing the number of lightpaths in the resulting logical topology. This traffic grooming problem is intractable even in simple network topologies, such as paths and stars, for which the routing and wavelength assignment (RWA) subproblem can be solved in polynomial time [13]. Consequently, for networks with more than a few nodes, it is important to develop heuristics which obtain good solutions in polynomial time.

Our framework for hierarchical traffic grooming 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 clusters consisting of a contiguous subset of nodes. One node within each cluster is designated as the *hub*, and is the only node responsible for grooming intra- and inter-cluster traffic. Hub nodes are provisioned with more resources (e.g., larger number of electronic ports and higher switching capacity) than non-hub nodes, and are similar in function to airports that serve as major hubs; these airports are typically larger than non-hub 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 [2]:

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. We describe in detail our clustering algorithm for traffic grooming in Section 3.
2. **Hierarchical logical topology design and traffic routing.** The outcome of this phase is a set of lightpaths for carrying the traffic 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) intra-cluster traffic grooming; and (c) inter-cluster traffic grooming. This hierarchical approach is described in detail in [2].

3. **Lightpath routing and wavelength assignment (RWA).** The RWA problem has been studied extensively in the literature. In this work, we adopt the LFAP algorithm [17], 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.

3 Clustering for Hierarchical Grooming

Clustering is a function that arises frequently in problems related to network design and organization. Clustering algorithms are classified as either *minimum cut* or *spanning tree*, depending on the underlying methodology [9]. The input to the algorithms generally consists of a set of nodes and edge weights, 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 topology; also, unlike typical objective functions considered in the literature, ours cannot be easily expressed as a function of the resulting clusters. Therefore, existing clustering techniques such as TPABLO [3] or METIS [16], are not directly applicable to the problem at hand.

One clustering problem that may be relevant to hierarchical traffic grooming is *K-Center* [8,11]. The goal of *K-Center* 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. The set S implicitly defines K clusters with corresponding hub nodes in S. *K-Center* is known to be NP-Complete [8]. We implemented the 2-approximation algorithm in [8], and compare it to our own in Section 5.

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 [7], and in [4], 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.

3.1 Important Considerations

We now discuss the tradeoffs involved in selecting the clusters, which set the design principles for the clustering algorithm we present in the next subsection.

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. This is a complex task as it depends on both the physical topology and the matrix T. Consider the tradeoffs involved in determining the number m of clusters. If m is small, the amount of inter-cluster traffic will

likely be large. Hence, the m hubs may become bottlenecks, resulting in a large number of electronic ports at each hub and a large number of wavelengths to carry the lightpaths over the links to/from each hub. On the other hand, a large m implies small clusters. In this case, the amount of intra-cluster 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 inter-cluster traffic. Therefore, one must select the number and size of clusters to strike a balance between capacity utilization and number of lightpaths for both intra- and inter-cluster traffic.

Now consider the composition of each cluster. If the average traffic demand between nodes within a cluster is higher than the average inter-cluster demand, there will tend to be fewer inter-cluster 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 RWA phase).

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 inter-cluster traffic, it is important to select clusters so that their cut size is not too small, so as 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. Creating clusters that consist of nodes on different sides of the cut is undesirable, because it may generate unnecessary traffic that goes back and forth through the cut. This traffic can be eliminated by forcing nodes on different sides of the bisection to be in different clusters. In Section 3.3, we describe a pre-cutting 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. In general, cluster topologies with relatively short diameter are more attractive in terms of RWA.

3.2 The MeshClustering Algorithm

Figure 1 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. The algorithm treats these parameters as an *indication* of the desirable range of cluster sizes, rather than as hard thresholds that cannot be violated. Consequently, the final result may contain clusters larger than *MaxCS* (see the discussion below regarding Step 24).

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-inter-cluster 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.

Procedure **MeshClustering**
begin
1. **while** $V \neq \phi$ **do**
2. $v \leftarrow$ node in V with max remaining capacity; $V \leftarrow V - \{v\}$
3. $B \leftarrow \{v\}$ // new cluster B with hub v
4. **while** $V \neq \phi$ and $|B| < MaxCS$ **do** // grow cluster B
5. $Q \leftarrow$ set of nodes $\in V$ adjacent to nodes in B
6. **foreach** node $q \in Q$ **do**
7. $B' \leftarrow B \cup \{q\}$ // assume q is included in B
8. HUBTEST: is traffic between B', $\overline{B'} > \Delta\times$ remaining hub capacity?
9. CUTTEST: is traffic between B', $\overline{B'} > \Delta\times$ remaining cut link capacity?
10. **if** q passes both tests **then**
11. $x \leftarrow$ total traffic between q and nodes in B
12. $y \leftarrow$ total traffic between q and nodes in $\overline{B'}$
13. $\rho_q \leftarrow x/y$ //intra- to inter-cluster traffic ratio
14. $d \leftarrow$ diameter of induced subgraph B'
15. $\delta_q \leftarrow d/|B'|$ // diameter-to-nodes ratio
16. **else** $Q \leftarrow Q - \{q\}$
17. **end for**
18. **if** $Q = \phi$ **then break** // cannot grow cluster B
19. **else**
20. $q_0 \leftarrow$ node $\in Q$ with largest ρ_q and smallest δ_q
21. $B \leftarrow B \cup \{q_0\}$; $V \leftarrow V - \{q_0\}$ // grow cluster B to include q_0
22. **end while** // continue until B cannot grow further
23. **end while**
24. Combine clusters of size $< MinCS$ with adjacent clusters
end

Fig. 1. Clustering algorithm for mesh networks

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 carry the traffic demands. Specifically, we require that the inter-cluster traffic originating from or terminating at a given cluster do not exceed a fraction Δ of the hub capacity (this is the HUBTEST in Step 9); similarly, this intra-cluster 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). The algorithm will consider a node to add to a cluster only if doing so will not violate these two constraints.

The parameter δ controls the ratio of the diameter of a cluster to the number of nodes it contains. To avoid cluster topologies that resemble long paths, we require that $0 < \delta \leq 0.75$. We used the value $\delta = 0.75$ which corresponds to a 4-node path, hence restricting the longest path within a cluster to at most three links. The parameter $\rho, 0.8 \leq \rho \leq 1.25$, specifies the acceptable range for the ratio of intra- to inter-cluster traffic for a cluster. Since 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, we used $\rho = 1.25$.

The MeshClustering algorithm in Figure 1 generates one cluster during each iteration of the main **while** loop between Steps 1 and 23. In Steps 2-3, 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 (refer to the second phase of the hierarchical grooming approach in Section 2). The cluster grows by adding one node during each iteration of the **while** loop between Steps 4 and 22. At each iteration, the set Q of candidate nodes for inclusion in cluster B consists of all nodes, not yet assigned to another cluster, which 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 8) and CUTTEST (Step 9); if not, node q is removed from consideration for inclusion into cluster B (Step 16). For all nodes q that pass both tests, we compute the diameter-to-nodes ratio δ_q and intra-to-inter-cluster traffic ratio ρ_q, assuming that q is added to cluster B (Steps 10-15). 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 to cluster B (Steps 20-21), and the process is repeated. If clusters with fewer than *MinCS* nodes are created, Step 24 removes them and includes their nodes into adjacent clusters. As a result, at the end some clusters may contain more than *MaxCS* nodes.

3.3 Pre-Cutting for Imbalanced Topologies

As we mentioned in Section 3.1, when the topology has a bisection of small cut size, the cut links are likely to become congested. Hence, it may be necessary to disallow nodes on different sides of such a bisection from being in the same cluster. However, identifying such a critical bisection in a large, imbalanced topology, is a difficult task. To tackle it, we use the CHACO software [10] which implements the partitioning algorithm in [14]. The software uses the parameter KL-IMBALANCE to control the relative sizes of the node sets on either size of the bisection. We apply CHACO several times, varying the KL-IMBALANCE parameter, and obtain several different bisections of the physical topology. We then select the bisection with the most traffic flowing along the cut links.

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.

4 Lower Bounds

A simple lower bound on the number of lightpaths (our main objective) can be calculated 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. However, we obtain a better lower bound based on the following observations. Let b_{sd} denote the number of direct lightpaths set up from s to d. Since all traffic originating (respectively, terminating) at node s (respectively, node d) must be carried on some lightpath also originating (respectively, terminating) at s (respectively, d), the following constraints must be observed:

$$\sum_d b_{sd}C \geq \sum_d t^{(sd)} \quad \forall s \qquad \sum_s b_{sd}C \geq \sum_s t^{(sd)} \quad \forall d \tag{1}$$

We can obtain a lower bound on the number of lightpaths by solving the ILP:

Minimize $\sum_{s,d} b_{sd}$ subject to constraints (1).

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 we consider in the next section. Although this bound is better than the bound above, we believe that it is somewhat loose.

For a lower bound on the number of wavelengths, consider the bisection cut of the network we identify using the approach described in Section 3.3. Let t be the maximum amount of traffic that needs to be carried on either direction of the links in the cut set. Also, let x be the number of links in the cut set, and C the capacity of each wavelength. Then, the quantity $\lceil t/xC \rceil$ is a lower bound on the number of wavelengths for carrying the given traffic matrix.

5 Numerical Results

We now present experimental results for two network topologies: a 47-node, 96-link network [1] with a balanced topology (i.e., there is no bisection with a small cut size that can be bottleneck in traffic grooming), and a 128-node, 321-link network which corresponds to the worldwide backbone of a large service provider (http://www.caida.org). The latter topology is imbalanced, as there exists a bisection with a cut size of 5 links that divides it into two parts of 114 and 14 nodes, respectively, hence we use the approach we described in Section 3.3 to create clusters that contain nodes on one side of the cut only.

The traffic matrix of each problem instance is generated by drawing $N(N-1)$ random numbers (rounded to the nearest integer) from a Gaussian distribution whose mean and standard deviation depend on the traffic pattern. We consider three patterns: *random*, which is challenging since the matrix does not have any structure that can be exploited by a grooming algorithm; *falling*, which is such that the amount of traffic between two nodes decreases with the distance between them; and *rising*, which is the opposite of the falling pattern.

For a given topology and traffic pattern, we generate thirty problem instances and we compare our MeshClustering algorithm to the *K-Center* algorithm [8]. We consider two performance metrics: the *normalized lightpath count* and the *normalized wavelength count*. The former is the ratio of the number of lightpaths required for hierarchical traffic grooming with one of the clustering algorithms,

to the lightpath lower bound of Section 4; the latter is the ratio of the number of wavelengths required to the wavelength lower bound of Section 4.

Figure 2 plots the normalized lightpath and wavelength count for each of thirty problem instances with a falling traffic 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. To make the comparison against *K-Center* as fair as possible, we selected two sets of values for the user-defined parameters of MeshClustering so that the average number of clusters over all thirty instances is 3.52 and 5.45, respectively.

From the figure, 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. As we mentioned earlier, we believe that this lower bound is rather loose, hence the performance of hierarchical grooming is better than the curves imply. Also, 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, note that a 1% reduction in the number of lightpaths in this network would result in about 40 fewer electronic ports, a substantial savings in cost.

We also observe that the MeshClustering algorithm requires significantly fewer wavelengths than the *K-Center* algorithm. This result is due to the fact that our algorithm is designed to take the wavelength requirements into account. In absolute terms, the difference in the number of wavelengths for these problem instances is in the order of 10-12. 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 cut of small size and large cross-cut traffic, but this 47-node network does not have such a bottleneck cut. Furthermore, the falling pattern makes it unlikely that a large amount of traffic will cross any cut.

Figure 3 is similar to Figure 2 but shows results for the rising pattern. Due to the nature of this pattern, relatively large amounts of traffic will cross any network cut, resulting in the much tighter wavelength bounds observed. Again, except for a few instances, our clustering algorithm outperforms the *K-Center* algorithm. We also observe that hierarchical grooming provides good solutions regardless of the clustering algorithm.

Finally, Figure 4 presents results for the 128-node network and the random traffic pattern. 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). Our clustering algorithm slightly outperforms *K-Center* in terms of the number of lightpaths, and both are relatively close to the lower bound. However, in terms of wavelengths, our algorithm produces results that are within 5% of the lower bound, whereas *K-Center* requires twice that number of wavelengths.

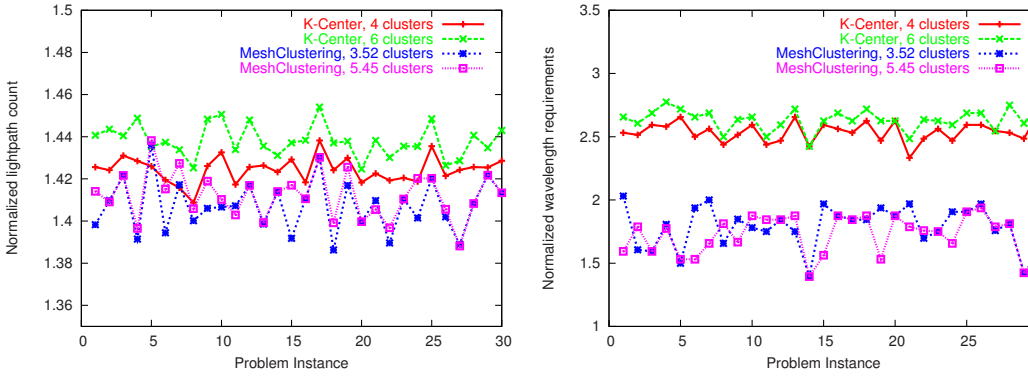

Fig. 2. Lightpath (left) and wavelength (right) comparison, falling pattern, 47-node network

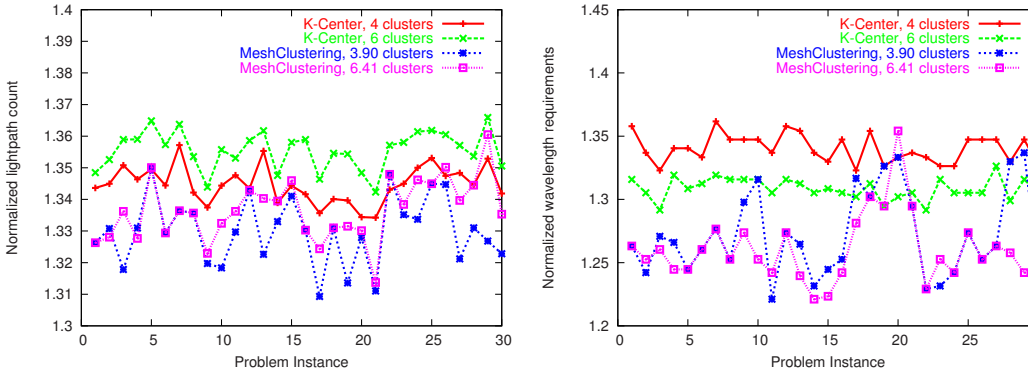

Fig. 3. Lightpath (left) and wavelength (right) comparison, rising pattern, 47-node network

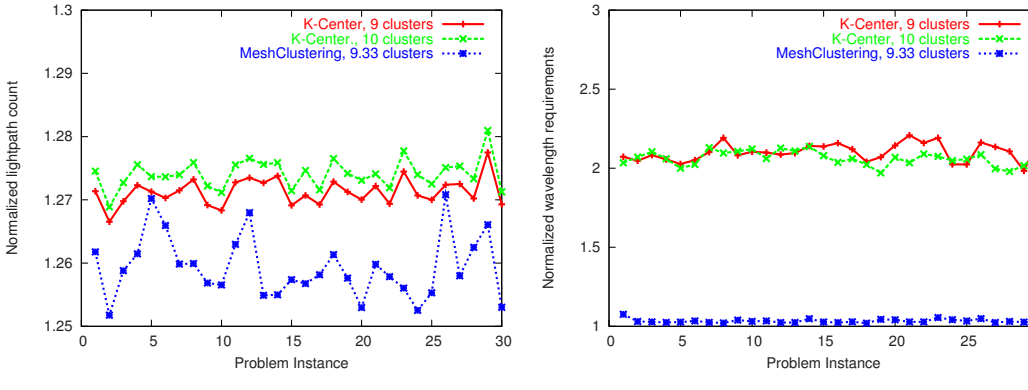

Fig. 4. Lightpath (left) and wavelength (right) comparison, random pattern, 128-node network

6 Concluding Remarks

Hierarchical traffic grooming is a new approach for efficient and cost-effective design of large-scale optical networks with multigranular traffic. The hierarchical grooming framework must be coupled with clustering techniques that follow grooming-specific design principles, and we have presented such a clustering algorithm that is flexible in balancing various conflicting goals via user-defined parameters. The experimental results demonstrate that (1) our hierarchical approach scales to large networks; (2) our clustering algorithm outperforms an algorithm that was not developed with traffic grooming in mind; and (3) hierarchical grooming combined with specially designed clustering techniques produce logical topologies that perform well across a variety of traffic patterns.

References

1. P. Baran. On distributed communications networks. *IEEE Trans. Commun.*, 12(1):1-9, Mar. 1964.
2. B. Chen, G. N. Rouskas, and R. Dutta. A framework for hierarchical grooming in WDM networks of general topology. *BROADNETS 2005*, pp. 167-176, Oct. 2005.
3. H. Choi and D. B. Szyld. Application of threshold partitioning of sparse matrices to markov chains. In *IPDS*, 1996.
4. Z. Ding and M. Hamdi. Clustering techniques for traffic grooming in optical WDM mesh networks. In *GLOBECOM 2002*, vol. 3, pp. 2711-2715, Nov. 2002.
5. R. Dutta and G. N. Rouskas. On optimal traffic grooming in WDM rings. *IEEE Journal on Selected Areas in Communications*, 20(1):110–121, January 2002.
6. R. Dutta and G. N. Rouskas. Traffic grooming in WDM networks: Past and future. *IEEE Network*, 16(6):46–56, November/December 2002.
7. M. Esfandiari, *et al.* Improved metro network design by grooming and aggregation of STS-1 demands into OC-192/OC-48 lightpaths. In *NFOEC 2003*, Sep. 2003.
8. T. Gonzalez. Clustering to minimize the maximum inter-cluster distance. *Theoret. Comput. Sci.*, 38:293–306, 1985.
9. John A. Hartigan. *Clustering Algorithms.* Wiley, 1975.
10. B. Hendrickson and R. Leland. The Chaco user's guide. Technical Report SAND95-2344, Sandia National Laboratories, Albuquerque, NM 87185-1110, July 1995.
11. D. Hochbaum and D.B. Shmoys. A best possible heuristic for the k-center problem. *Math. Oper. Res.*, 10:180–184, 1985.
12. J.Q. Hu and B. Leida. Traffic grooming, routing, and wavelength assignment in optical WDM mesh networks. *INFOCOM 2004*, 23(1):495-501, Mar. 2004.
13. S. Huang, R. Dutta, and G. Rouskas. Traffic grooming in path, star, and tree networks: Complexity, bounds, and algorithms. *JSAC*, 24(4):66-82, Apr. 2006.
14. B. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal*, 29:291–307, 1970.
15. V. R. Konda and T. Y. Chow. Algorithm for traffic grooming in optical networks to minimize the number of transceivers. In *HPSR Workshop*, pp. 218-221, 2001.
16. K. Schloegel, G. Karypis, and V. Kumar. A new algorithm for multi-objective graph partitioning. TR99-003, Dept. of CS, Univ. of Minnesota, Sep. 1999.
17. H. Siregar *et al.* Efficient routing and wavelength assignment in wavelength-routed optical networks. *7th Asia-Pacific NOMS*, pp. 116-127, Oct. 2003.
18. H. Zhu *et al.* A novel generic graph model for traffic grooming in heterogeneous WDM mesh networks. *IEEE/ACM Trans. Netw.*, 11(2):285-299, Apr. 2003.