Grooming of Non-Uniform Traffic on Unidirectional and Bidirectional Rings*

Raza Ul-Mustafa† Ahmed E. Kamal‡

Department of Electrical and Computer Engineering
Iowa State University
Ames, IA 50011-3060
U.S.A.
{raza,kamal}@iastate.edu

Abstract

Traffic grooming in WDM networks is obtained by intelligently allocating the traffic onto a given set of wavelengths. This paper presents heuristics for grooming of non-uniform general traffic demands onto a given set of wavelengths available on a unidirectional or bidirectional ring. The objective is to minimize the number of higher layer equipment, like SONET Add/Drop Multiplexers (ADMs), or MPLS routers. We map the unidirectional ring onto a linear topology and develop a generalized two-step approach to solve the grooming problem on the mapped topology. In the first step, we allocate the traffic while minimizing the possible number of *strings* (each string being a collection of non-overlapping traffic streams) in a manner that yields the optimal number of strings in the linear topology case. We also prove the optimality of this step in the number of the strings (wavelengths). In the second step we employ a grouping technique to efficiently combine g strings onto a wavelength while minimizing the total number of the ADMs. We also address the problem of grooming the non-uniform traffic on a bidirectional ring by mapping it onto unidirectional rings, and applying the two-step approach. Moreover, in the case of bidirectional rings we propose an approach to route the traffic that reduces the total number of the required wavelengths and ADMs. The time complexity of our technique is at least an order of n less than other proposed approaches, where n is the total number of nodes in the network. The efficacy of the proposed technique has been demonstrated through a large number of experiments.

Keywords: WDM networks, traffic grooming, ADMs, linear topology, unidirectional ring, bidirectional ring, nonuniform traffic, asymmetric traffic, NP-Completeness

*This research was supported by grant ANI-0087746 from the National Science Foundation.

†R. Ul-Mustafa is now with Microsoft Corporation, One Microsoft Way, Redmond, WA 98052-6399, U.S.A.; e-mail: razaulm@microsoft.com.

‡Corresponding author.

I Introduction

During the last decade, Wavelength Division Multiplexing (WDM) Networks have emerged as an attractive architecture for backbone networks. WDM networks provide high aggregate bandwidth, on the order of several Terabits per second. Also, WDM networks eliminate the electro-optic processing delays using wavelength routing [1]. However, the cost-effectiveness of WDM networks depends on the amount of the optical passthrough provided by the network to the given traffic. The amount of the optical passthrough in turn depends on the traffic pattern and on the way the traffic between different source and destination pairs is groomed (multiplexed) on the available set of wavelengths. Traffic grooming is thus defined as an intelligent allocation of the traffic demands, between different network nodes, onto an available set of wavelengths in such a way that reduces the overall cost of the network.

WDM routing networks support lightpaths, which is a pure optical communication path between two nodes. In order to optimize the cost of the network, one needs to take into account the higher layer that will use these lightpaths and its connectivity patterns. The Synchronous Optical Networks (SONET) is currently being used as a higher layer in WDM networks, and because of its wide deployment and efficient protection schemes will remain the most likely option for some time.

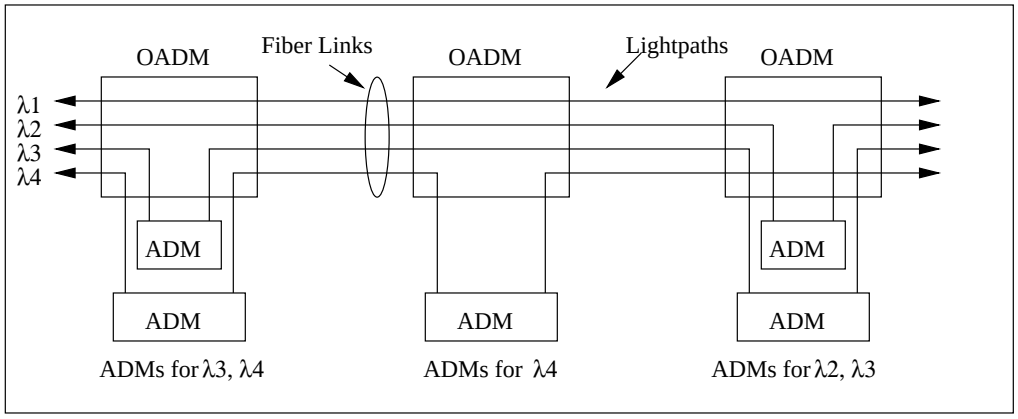

Figure 1: An example of an optical network with OADMs and ADMs.

In Figure 1, a typical WDM network is shown with three nodes. Each node is equipped with an Optical Add/Drop Multiplexer (OADM), which can selectively add or drop wavelengths at each node, thus providing an optical passthrough to the rest of the wavelengths. Each of the wavelengths dropped at a node is then processed by the higher layers Add/Drop Multiplexers (ADMs), e.g., a SONET ADM, after conversion into the electronic form. Thus the equipment needed at each

node corresponds to the number of wavelengths dropped at each node. To reduce this number, and consequently the cost of equipment, one needs to reduce the number of wavelengths dropped at a node. We can achieve this goal by grooming the traffic in such a way that all the traffic to and from a node is carried on the minimum number of wavelengths.

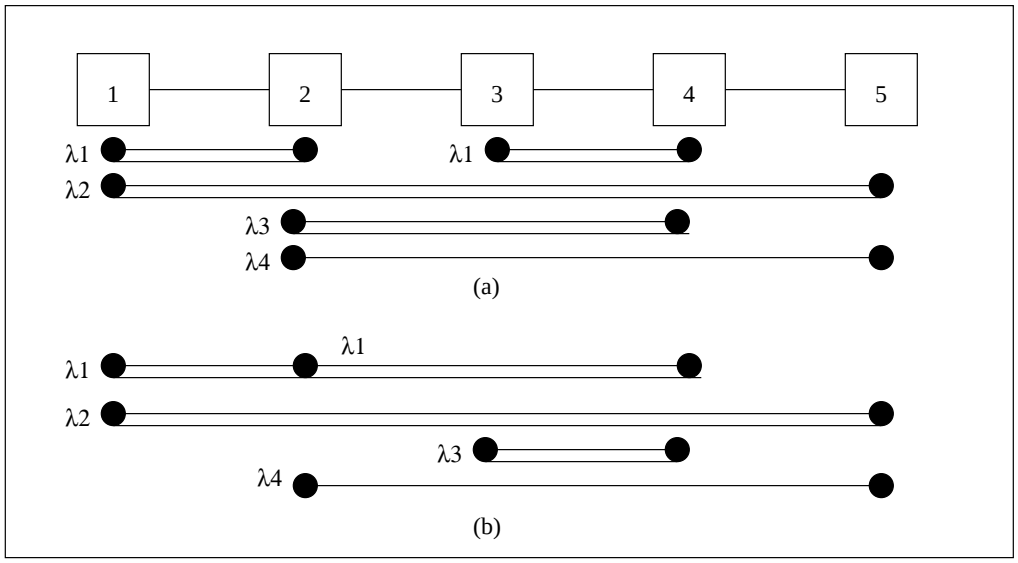

Figure 2: Two different traffic assignments on a linear topology, for the same traffic demands.

As an example, consider a 5-node linear topology network, shown in Figure 2. Let each node be equipped with an OADM. Each OADM is in turn connected to a number of SONET ADMs (not shown in the figure). Having an OADM on each node will help only drop those wavelengths that carry the traffic to, or from that specific node. A wavelength can bypass a node if it carries no traffic that is transmitted or received by that node. This will result in the saving of a SONET ADM. Hence, our objective is to minimize the total number of the SONET ADMs used in the network to support all of the traffic by intelligently assigning traffic to the wavelengths. Let g denote the total number of basic units of traffic supported by each wavelength. For example, if a wavelength supports an OC-12 connection, and the basic unit of traffic is an OC-3, then $g = 4$. For illustrative purpose, we assume that $g = 2$ in Figure 2. Traffic demands or connections are shown by the line segments between the source and destination nodes. Also, black circles are used on the edges of the segments to represent a SONET ADM that is used at the corresponding node, and for a specific wavelength. To minimize the total number of ADMs required, out of many possibilities, we could have the following two allocations of traffic to wavelengths.

(a) λ_1: $1 \leftrightarrow 2$, $3 \leftrightarrow 4$; λ_2: $1 \leftrightarrow 5$; λ_3: $2 \leftrightarrow 4$; λ_4: $2 \leftrightarrow 5$;

(b) λ_1: 1↔2, 2↔4; λ_2: 1↔5; λ_3: 3↔4; λ_4: 2↔5;

Assignments (a) and (b) are shown in Figure 2(a) and 2(b), respectively. For the first assignment the total number of required ADMs is 10, while for the second assignment the total number of ADMs is 9. Note that in the second assignment the traffic between node 1 and 2, and between 2 and 4 are sharing the same ADM. This example shows that by assigning the traffic to appropriate wavelengths, one can reduce the number of required ADMs. Also, given very high cost of SONET ADMs, approximately $40,000 for a single port version, even saving few ADMs translates into savings of hundred thousands of dollars.

II Related work

Traffic grooming in WDM networks is comparatively a new field, and has recently started to receive attention. Few survey papers have been published in this area [3, 4, 5]. The general problem of traffic grooming with arbitrary traffic has been proven to be NP-Complete in [2]. Most of the work related to traffic grooming in WDM networks has an emphasis on reducing the number of required higher layer components [4]. However, there are some notable exceptions in which other factors, like the network utilization, are optimized, e.g., [18, 19]. Also, most of the work focuses on the special cases of the traffic and the specific network topologies. Many researchers focused on obtaining the upper and lower bounds on the number of the ADMs required for the specific topologies and traffic patterns [2, 9, 7, 6, 8], while few researchers proposed problem-specific heuristics [12, 13]. In the grooming literature, uniform traffic is defined as the same amount of traffic between all possible source-destination node pairs, while non-uniform traffic is defined as the variable amount of traffic between different source-destination node pairs. For traffic grooming problem, when traffic between different node pairs is non-uniform, few studies exist [8, 10, 12, 13]. Many of such studies consider only non-uniform symmetric traffic, i.e., traffic from(to) a specific source to(from) a specific destination is same.

In [2] the authors besides proving the NP-Completeness of the problem, also provided algorithms to minimize the number of the ADMs when the traffic from all nodes is destined to a single node and all traffic rates are the same. For the more general case of all-to-all uniform traffic, they obtained a lower bound on the number of the ADMs required and provide a heuristic to closely approach that bound. Finally, they also considered the use of a hub node, where the traffic can be switched between different wavelengths, and obtain an optimal algorithm that minimizes the number of the

ADMs by efficiently multiplexing and switching the traffic at the hub. In [6], the authors considered six different optical WDM ring architectures. They, then provide bounds on the number of the wavelengths, number of the ADMs and maximum hop length for each of the ring architectures. They also considered three different traffic models, namely, Static, Dynamic and Incremental, and calculated the bounds for each of the proposed six WDM ring architecture considering some specific traffic model. In [7], the authors while grooming the traffic considered the characteristics of the SONET UPSR and BLSR rings. They presented the lower bound on the number of the ADMs in a WDM UPSR ring, under the uniform traffic assumptions. Also they considered the single-hub UPSR WDM ring and obtained the bound on the number of ADMs required in case of uniform traffic. Similarly, they obtain the bounds on the number of ADMs required for BLSR/2 WDM rings under the uniform traffic assumption. In [8], the authors concentrated on the Single-Hub SONET/WDM ring architecture and obtained bounds on the number of the ADMs required under both uniform and non-uniform traffic. They reduce the single hub traffic grooming problem to the bin-packing problem, and hence used the approximation algorithms for bin-packing to solve the non-uniform traffic case in a near-optimal way. In [9], the authors also focused on the WDM rings and obtained the lower and the upper bounds by decomposing the ring into sets of nodes and adopting the locally optimal topology within each set. The optimal solutions obtained from sets of nodes are then combined to get a near-optimal solution for the whole ring. In [10], the authors consider the traffic scenario in which traffic streams can exist between any arbitrary pair of nodes. However, the demands between same node pairs are considered to be symmetric. In [12], the authors proposed few algorithms to achieve traffic grooming in SONET/WDM rings, under both uniform and non-uniform traffic. They followed a two-step approach, namely circle construction and circle grooming. In the circle construction phase they try to construct as few circles as possible to include all the requested connections; this helps to minimize the number of the wavelengths. In the circle grooming phase they try to combine the circles in such a way that the number of the ADMs used can be minimized. In [13], the authors improved the work done in [12] by using the simulated annealing to groom the circles.

Our contribution to the grooming problem is many-fold. We develop a generic model that can accommodate general non-uniform and asymmetric traffic with an arbitrary number of nodes and arbitrary grooming factor. We devise algorithms to solve the traffic grooming problem for our generic model. The proposed algorithms scale well with the problem size, and are efficient in terms of the solution quality, and run time. Another contribution of our work is that, in case of bidirectional

rings, we demonstrate that the shortest-path routing does not necessarily lead to minimizing the number of the wavelengths and the ADMs, and propose an approach to route the traffic in a way that reduces the total number of the required wavelengths and ADMs.

The rest of the paper is organized as follows. In Section III we map the unidirectional ring onto a linear topology and present the two-step approach that solves the grooming problem for non-uniform traffic. In Section IV we map the bidirectional rings onto a linear topology and also investigate the different routing strategies. Section V presents several experimental results, while Section VI concludes the paper.

III Traffic Grooming on Unidirectional Rings

In this section we define the terms used in the rest of the paper, and map the unidirectional ring onto a linear topology. We then develop a two-step approach that handles all types of traffic, uniform and non-uniform (including both symmetric and asymmetric), on the mapped topology. We also show that the first step of our approach is optimal in the number of the wavelengths for a linear topology.

Let the total number of nodes be N, and let the traffic matrix C be defined as, $C = [c_{ij}]$ such that $1 \leq i \leq N$ and $1 \leq j \leq N$. Each c_{ij} entry is a set that represents the traffic units between nodes i and j and consists of $|c_{ij}| = n_{ij}$ number of basic units of data, namely, $c_{ij} = \{c_{ij}^{(1)}, c_{ij}^{(2)}, ..., c_{ij}^{(n_{ij})}\}$. We call each such basic unit of data, $c_{ij}^{(k)}$, a *stream* or a *connection*. Let r represent the rate of the basic stream, e.g., an OC-3 connection. Our objective is to accommodate the demands in the traffic matrix C with the least possible number of wavelengths (W), and number of SONET ADMs (D). In general, by reducing W, we will be able to reduce D, since for each originating and terminating wavelength at a node (in the same direction) we need an ADM. This, however, also suggests that to reduce D, besides reducing W, we need to allocate most of the traffic to and from a node on as few wavelengths as possible.

To determine the lower bound on the number of wavelengths we will define the term *density* (d) as the maximum number of streams on any of the N links. Let link l be the link between nodes l and $(l+1) mod N$. Then density d_l at link l can be defined as:

$$d_l = \sum_{i \leq l, j > l} n_{ij} \tag{1}$$

Similarly, the *density* can be defined at a node. Let d_i stand for the density at node i, and T_i,

S_i, and P_i stand for the number of terminating streams at node i, the number of starting streams from node i, and the number of streams that are passing through node i, respectively. Then:

$$d_i = \max(T_i, S_i) + P_i \tag{2}$$

$$d = \max_i d_i \quad 1 \leq i \leq N \tag{3}$$

The density at a node or link shows that to accommodate the traffic, we need at least this much bandwidth (in terms of the number of streams). Therefore, the minimum number of wavelengths can be determined as:

$$W_{LB} = \left\lceil \frac{d}{g} \right\rceil \tag{4}$$

To determine a tight lower bound on the number of ADMs for non-uniform traffic is, however, quite difficult. In [10], the authors generalized the work done by [14], by incorporating the grooming factor, as follows.

$$D_{LB} = \sum_{i=1}^{N} \left\lceil \frac{\max(T_i, S_i)}{g} \right\rceil \tag{5}$$

In [11], the authors proposed an even tighter bound for circular rings as follows.

$$W_{LBring} = \sum_{i=1}^{N} (T_i + S_i - m_i) \tag{6}$$

where m_i is the size of the maximal matching of the bipartite graph, constructed at each node, of starting and terminating streams as vertices, and edges correspond to non-overlapping streams.

III.1 Mapping of a Unidirectional Ring onto a Linear Topology

In this subsection we devise an innovative method to map the unidirectional ring onto the linear topology.

In a unidirectional ring of N nodes, let the nodes be numbered from 1 to N, starting from any node, in the direction of communication. Then, the traffic from node i, where $1 < i \leq N$, destined to node j, where $j < i$, must be crossing the links between nodes N and j. By adding $N-1$ dummy nodes to an N node linear topology, we can emulate the behavior of an N node unidirectional ring. All the traffic sourced by i and destined to j, for $j < i$ and $i > 1$, on a unidirectional link, will now be terminating at node $N + j$.

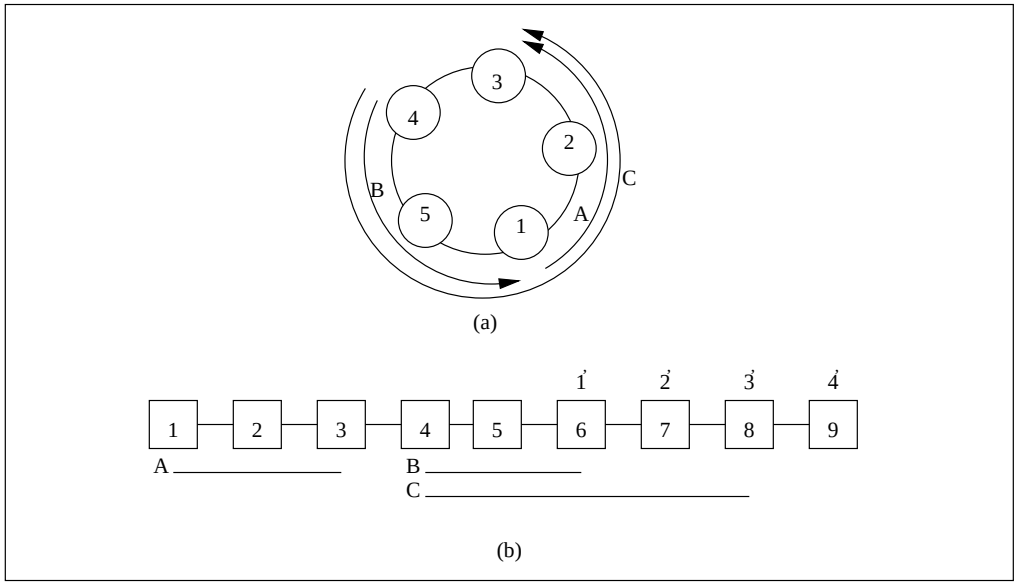

Figure 3: Mapping a unidirectional Ring into linear topology. Nodes 6 to 9 are the added dummy nodes corresponding to nodes 1 to 4.

In Figure 3, a simple 5 node unidirectional ring is shown. Also the unfolding of the ring into a linear topology is depicted. Nodes 6, 7, 8, and 9 (also represented as 1', 2', 3', and 4 ', respectively) are the added dummy nodes, which correspond to nodes 1, 2, 3, and 4, respectively. Traffic sourcing from a node and destined to a lower indexed node will now terminate at the corresponding added dummy node. For example, traffic originating at node 4 and terminating at destination 1, will now terminate at node 6, as shown in Figure 3. Once all the traffic is mapped from a unidirectional ring onto a linear topology, the solutions developed for the linear topology below will be applicable to the unidirectional ring too.

III.2 A Two Step Approach

In this sub section, we present a two step approach to solve the grooming problem on a linear topology. In the first step, we devise an algorithm MIN-STRINGS that arranges a number of non-overlapping streams into a *string*, thus making *strings* of streams in such a way that minimizes the total number of strings. Each string will be formed such that the space between streams in the same string is minimized. This algorithm, optimally minimizes the number of strings, which is equivalent to the lower bound on wavelengths. In the second step, we use a grouping heuristic that groups g (or fewer) strings onto each wavelengths. During grouping, our objective is to group

strings together in a manner that results in reducing the number of the ADMs.

III.2.i Minimizing the Number of Strings

In this subsection we present a novel technique that accommodates the general non-uniform traffic and produces the least number of strings, equal to the density d, thus leading to the least number of wavelengths, W for a linear topology.

To approach the problem in hand, a visualization of the problem will be helpful. Considering Figure 2 again we notice that to compactly pack the traffic streams we can combine them horizontally as *strings* such that no two traffic streams in a string overlap. Formally, $G = (V, E)$, where V is the set of nodes and E is the set of edges, is an interval graph provided that one can assign to each an interval I_v such that $I_u \bigcap I_v$ is nonempty precisely when $u, v \in E$. Visualizing each stream as an interval we can then form an interval graph consisting of all the streams such that, each stream is represented by a vertex and there is an edge between each pair of vertices (intervals) that overlaps.

The problem of finding the minimum number of strings is then equivalent to finding a vertex coloring of the corresponding interval graph with a minimal number of colors [15]. The graph coloring problem for general graphs is NP-Complete [16]. However, for interval graphs we will present a polynomial time algorithm that is optimal in the number of colors (in our case each color corresponds to a *string*). For further information on interval graphs, coloring of interval graphs, and its complexity issues, please consult [15, 16, 17].

Let us represent each stream c_{kl} with a pair of coordinates $(Xmin(c_{kl}), Xmax(c_{kl}))$ such that $Xmin(c_{kl}) = k$ and $Xmax(c_{kl}) = l$. An i^{th} string, R_i, can then be defined as a set of non overlapping streams such that its member streams are not present in R_j, where $1 \leq j \leq |C|$, $j \neq i$, and $|C|$ is the total number of streams. From now on, we will use the terms *stream* and *segment* interchangeably.

Let L represent a sorted list of all the streams $c_{ij} \in C$, and $L(i)$ represent the i^{th} element in the list L. The sorting criterion is explained in the algorithm itself. Also let us define an operation $REMOVE(L(i))$ on list L that removes $L(i)$ from the list L and hence decrements its size, $|L|$, by one (for $i \geq 0$). The algorithm MIN-STRINGS is then given in Figure 4.

The algorithm MIN-STRINGS starts by sorting the segments in list L in ascending order with respect to their $Xmin$ coordinate. For segments having the same $Xmin$ coordinates, longer segments are selected first (for segments with the same $Xmin$ and length, the tie can be broken randomly). For each string we repeat the following (lines 3-18). We first initialize each string by the first element in the sorted list and remove that element from the list (lines 5-8). We then *fill* the string by

Algorithm MIN-STRINGS

input:Matrix C
output:A set of strings $R = \{R_1, R_2, ..., R_k\}$ whereas $k = |R|$

BEGIN
1. Sort all the streams $c_{ij} \in C$ in List L in ascending order with respect to the first index, such that c_{ij} precedes c_{kl} $\forall i < k$, and in descending order with respect to the second index, such that c_{ij} precedes c_{ik} $\forall j > k$.
2. $i \leftarrow 1$;
3. WHILE ($L \neq \emptyset$) DO
4. BEGIN
5. $q \leftarrow L(1)$;
6. $R_i \leftarrow q$;
7. $REMOVE(L(1))$;
8. $j \leftarrow 1$;
9. WHILE ($(L \neq \emptyset) \wedge (j \leq |L|)$) DO
10. BEGIN
11. IF($Xmax(q) \leq Xmin(L(j))$) THEN
12. $q \leftarrow L(j)$;
13. $R_i = R_i \cup q$;
14. $REMOVE(L(j))$;
15. ELSE $j \leftarrow j + 1$;
16. END WHILE
17. $i \leftarrow i + 1$;
18. END WHILE
END MIN-STRINGS

Figure 4: Algorithm MIN-STRINGS for arrangement of traffic steams onto fewest number of *strings*.

searching the whole list, and accommodating the very first non-overlapping segments (lines 9-15). This way, in a single scan of the list we will be able to *fill* a string with available non-overlapping segments.

The algorithm MIN-STRINGS is simple and elegant and works for any general non-uniform traffic. Also, it always finds the optimal (minimum) number of strings, equal to the density d, as proven by Theorem 1 below. Moreover, simulation results verify that the algorithm compactly packs the segments in each string, which means that most of the segments are connected to each other. This reduces the number of the ADMs since two connected segments use the same ADM at their connecting point.

We can determine the complexity of MIN-STRINGS as follows. For each string, MIN-STRINGS scans the whole list. As the number of the strings produced by MIN-STRINGS is equal to the

density d, the complexity of the algorithm is $O(d \times |C|)$. If the traffic between different node pairs is uniformly distributed between zero and some number h, then on average the total number of segments between all nodes pairs is $(N^2 \times \frac{h}{2})$, and the complexity of the MIN-STRINGS can be given as $O(d \times N^2 \times h)$.

Before proving the optimality of the MIN-STRING algorithm, we will define a few terms, followed by lemmas that are required for the proof.

Let nodes be numbered $1, 2, ..., N$, and the links be numbered $1, 2, ..., N-1$, such that link i is between nodes i and $i+1$. Let Ω be the set of all given segments, i.e., $\Omega = \{a_1, a_2, \ldots, a_{|C|}\}$. Let ψ_j be a set that consists of segment a_j and all segments that overlap with a_j and starts earlier than or at the same node as a_j, i.e., $\psi_j = \{x | x \in \Omega \wedge (Xmin(x) \leq Xmin(a_j)) \wedge (Xmax(x) > Xmin(a_j))\}$. Let π_i be a set defined as, $\pi_i = \{\psi_m | \sup_{Xmin(a_m) \leq i < Xmax(a_m)} m\}$. Basically, if a segment a_m starts at node i, then $\pi_i = \psi_m$. If no segment starts at node i then we will select such a segment a_m, which passes through node (link) i, and its starting point is closest to node i among all the segments passing through node (link) i. π_i will then consist of segment a_m and all the segments that overlap with a_m. Note that if no segment passes through link i, then π_i will be empty. Finally, let Π be a sequence of sets π_i, i.e., $\Pi = (\pi_1, \pi_2, ..., \pi_{N-1})$.

The following lemmas are needed for the proof of the theorem.

Lemma 1: $|\psi_j| \leq d$ for $1 \leq j \leq |C|$

Corollary: $|\pi_i| \leq d$ for $1 \leq i \leq N-1$

Lemma 2: In each iteration, k, of the MIN-STRINGS algorithm (lines 3-18, Figure 4), exactly one member of each nonempty $\pi_i, 1 \leq i \leq N-1$, will be selected for inclusion in string k.

The following theorem is the main result.

Theorem 1: MIN-STRINGS algorithm is optimal in the number of strings.

The proof of the lemmas and the theorem is given in the appendix.

Figure 5 shows the output of MIN-STRINGS for a sample input. The traffic demands are shown in Figure 5(a). Each segment corresponds to a single traffic unit. Figure 5(b) shows the demands after sorting. Notice that longer segments precede shorter segments when their $Xmin$ coordinates are same. Figure 5(c) shows the output of MIN-STRINGS. Notice that the number of strings is exactly equal to the density, which is 8 in this case.

Please note that on a unidirectional ring, the segments between original nodes and the added dummy nodes could be overlapping. Any segment crossing the link between nodes i and $i+1$, for $1 \leq i \leq N$, overlaps with any segment crossing the link between nodes $N+i$ and $N+i+1$. For

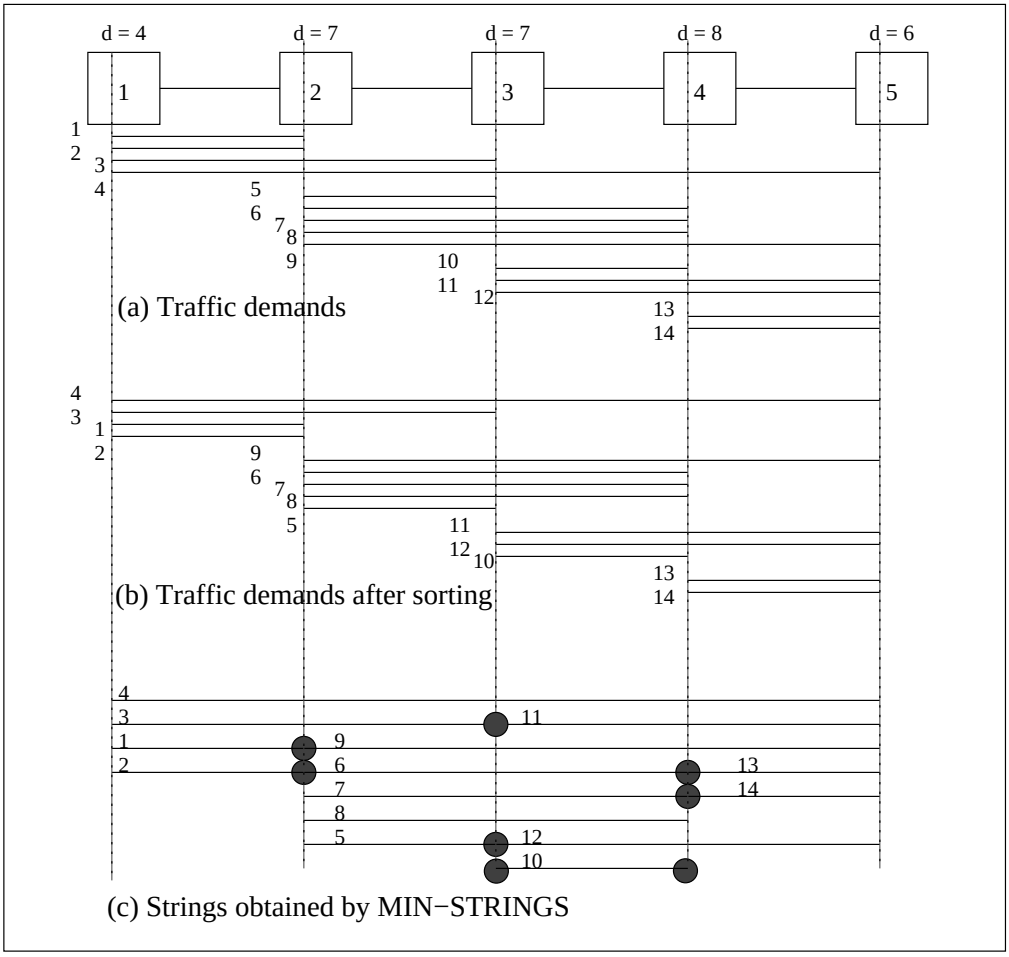

Figure 5: Output of the MIN-STRINGS algorithm for a sample set of traffic demands.

example, from Figure 3(b) it appears that segments A and C can be combined in a string. However, they are overlapping segments. The links between nodes 6 and 8 are the same as the links between nodes 1 and 3. Therefore, we need to modify a few expressions for unidirectional ring. Let $d_{i,i+1}$ be the density of the link between nodes i and $i+1$. For a unidirectional ring, $d_{i,i+1}$ can be defined as:

$$d_{i,i+1} = \sum_{k=1}^{i} \sum_{l=i+1}^{N} n_{kl} + \sum_{k=i+2}^{N} \sum_{l=i+1}^{k-1} n_{kl} \tag{7}$$

The first term includes all the sources before and including node i, and the destinations after and including node $i+1$. The second term considers all the sources k between $i+2$ and N, that are transmitting the traffic either to node $i+1$, or to nodes after $i+1$ but before node k.

Similarly, we need to make few minor modifications in the algorithm MIN-STRINGS. While packing each string we need to select only non-overlapping segments. Therefore, besides checking the condition, $Xmax(A) \leq Xmin(B)$ (line 11 in MIN-STRINGS), we also need to check $Xmin(A) \geq$

$Xmax'(B)$, where $Xmax'(B)$ is defined as:

$$Xmax'(B) = \begin{cases} 0 & 1 \leq Xmax(B) \leq N \\ Xmax(B) mod N & Xmax(B) > N \end{cases}$$

Note that enforcing the above constraint will take into account the actual physical capacity of a node. Basically, here we are making sure that each string consists of only non-overlapping segments. Later, in a grouping algorithm we will combine at most "g" number of such strings per wavelength, thus never exceeding the physical capacity of a node.

The number of strings determined by MIN-STRINGS in this case, will not necessarily be equal to the density. In fact, the problem of finding minimum number of strings in rings can be reduced to a circular-arc coloring problem [11], which is an NP-Complete problem [16]. The experimental results, however, show that the number of strings determined by MIN-STRINGS even in this case is quite close to the density. Note that we are opening up the ring at a particular node only. Opening the ring at different locations can affect the results of MIN-STRINGS. The distribution of the traffic between actual node and its dummy node could also affect the total number of strings determined by MIN-STRINGS. For example, in Figure 3, out of a total of h streams, say, between node 2 and 4, a fraction of traffic streams can be allocated between node 7 and 9, thus influencing the output of MIN-STRINGS. One of the possibility is to open the ring at each node, each time extending the ring as explained above, and selecting the best solution. However, this will increase the complexity of the technique by an order of N. In Section V, we will examine the trade-off between number of ADMs and the corresponding time complexity of the heuristic, for the unidirectional ring.

III.2.ii Grouping Algorithm

By now we have the set of strings, R, of size $|R|$. In this subsection we will combine the $|R|$ strings to obtain W wavelengths, as given by equation [4]. Also while combining the strings, our objective will be to minimize D.

Before presenting the algorithm for grouping we define a few terms. Define S_{R_i} as the set of node points for which string i needs an ADM. In general each segment needs an ADM at its $Xmin$ and $Xmax$ positions. However, two connected segments on the same string can share a single ADM, because $Xmax$ of one of the segments is the same as $Xmin$ of the other segment. Please also note that in the mapped linear topology having any $Xmin$ or $Xmax$ at node i $(1 \leq i \leq N)$ is equivalent to having it on node $N + i$, and vice versa. We also define a *saving* function between strings i and

j as:

$$Saving(R_i, R_j) = \left| S_{R_i} \cap S_{R_j} \right| \tag{8}$$

So the *saving* of two strings depends on the number of node points, where both strings need an ADM due to the overlap of their component segments' endpoints ($Xmin$ and/or $Xmax$). Maximizing the *saving* function increases the sharing of ADMs, resulting in fewer ADMs. Also we define an operation MERGE on any two strings R_i and R_j as follows:

$$MERGE(R_i, R_j) \equiv S_{R_i} \leftarrow S_{R_i} \cup S_{R_j} \tag{9}$$

Thus the MERGE operation basically superimposes those node points of a string onto another where an ADM is needed. Let the list L now include all the elements of the set of strings R in the order of their creation by MIN-STRINGS. The grouping algorithm given in Figure 6 is then executed.

Algorithm GROUPING

input:List L consisting of all the elements, $R_1, R_2, \ldots, R_k$, in the set R.
output:A set of wavelengths $\lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_W\}$ and corresponding number of ADMs

BEGIN
1. $i \leftarrow 0$;
2. $D \leftarrow 0$;
3. WHILE ($L \neq \emptyset$) DO
4. BEGIN
5. $i \leftarrow i + 1$;
6. $\lambda_i \leftarrow L(1)$;
7. $G \leftarrow L(1)$;
8. $REMOVE(L(1))$;
9. FOR $j = 2$ to g DO
10. BEGIN
11. $k^* = \{k \mid Saving(G, L(k)) \geq Saving(G, L(l)), for 1 \leq l, k \leq |L|\}$;
12. $\lambda_i \leftarrow \lambda_i \cup L(k^*)$;
13. $MERGE(G, L(k^*))$;
14. $REMOVE(L(k^*))$;
15. END FOR
16. $D \leftarrow D + |S_G|$;
17. END WHILE
END GROUPING

Figure 6: Algorithm to group the *strings* obtained by MIN-STRINGS into W wavelengths while minimizing the number of the required ADMs.

The algorithm GROUPING starts with a list consisting of all the elements of the set R. Each wavelength is initialized with the very first string in the list L (line 6). The added string is removed from L (line 8). Next, such a string is selected from all the remaining strings that has maximum common node points, thus maximizing the *saving* (line 11). The largest value of *saving*, corresponding to a pair of strings, indicates that grouping these strings together in a wavelength will lead to sharing the largest number of ADMs. The selected string is then merged with the previously selected strings on the same wavelength (line 13). For each wavelength we group g strings. However, note that the number of strings selected for the W^{th} wavelength may be fewer than g. The number of ADMs required for each wavelength is then the number of $Xmin$ and $Xmax$ points over all the segments in the wavelength, while excluding the multiplicity of common node points. Note that we are allowing traffic between same pair of nodes to be allocated to different wavelengths.

The complexity of the algorithm GROUPING is $O(|R|^2)$. For the linear topology, $|R| = d$, and hence the complexity for linear topology can be expressed as $O(d^2)$. As $|R|$ is upper bounded by the total number of segments $N^2 \times \frac{h}{2}$ (an extreme case when each string has just one segment), hence the dominant factor in the overall complexity of the two-step approach comes from MIN-STRINGS that is $O(d \times N^2 \times h)$.

IV Traffic Grooming on Bidirectional Rings

In this section we show how to groom the non-uniform traffic on a bidirectional ring by mapping it onto unidirectional rings, which in turn will be mapped onto the linear topology. Also, we explore two segment routing options, namely, when the segments are routed using the shortest path, and when the route is not fixed and thus may or may not follow the shortest path route.

We first start with the shortest path approach. First, note that similar to the case of the unidirectional ring, the linear topology can also emulate a bidirectional ring by adding some dummy nodes. However, this time the number of added dummy nodes is just $\lfloor \frac{N}{2} \rfloor$. The allocation of the traffic streams, between source s and destination d, to the extended linear topology then can be explained as follows. When $(d > s)$, we will schedule the traffic from s to d if $((d - s) < \frac{N}{2})$ and from $s + \frac{N}{2}$ to d if $((d - s) > \frac{N}{2})$. In case when $((d - s) = \frac{N}{2})$, we will split the traffic into two halves, and assign each half from s to d and from $s + \frac{N}{2}$ to d. Also in case when $(d < s)$ and $((d - s) > \frac{N}{2})$ we will route the traffic from s to $d + \frac{N}{2}$. Note, however, that above assignment on a single ring can lead to segments in both directions. Applying MIN-STRINGS may results in combining opposite

direction segments into a string, thus implying that a bidirectional ADM (on a single wavelength) exists. To avoid such an assumptions and to consider ADMs that operate only in one direction, we need to modify our model. One of necessary update is to consider that each wavelength can contain streams *flowing* in one direction only. Groups of such g wavelengths then can be assigned to two different fibers (in case of unidirectional fiber) or to same fiber (in case of bidirectional fiber). Let the total traffic on a bidirectional ring be split between two unidirectional rings A and B. Also, let the corresponding extended linear topology, of ring A and B be represented as $L(A)$ and $L(B)$, respectively. The assignment of traffic to $L(A)$ and $L(B)$ can then be carried out as explained in algorithm ASSIGNMENT, given in Figure 7.

```
Algorithm ASSIGNMENT

input:traffic matrix C
output:An assignment of traffic streams on L(A) and L(B)

BEGIN
WHILE( c_sd > 0 ) ∀s, d DO
BEGIN
   IF(d > s) THEN
      IF((d − s) > N/2) THEN
         Route traffic from s + N to d on L(B)
      ELSE IF( (d − s) = N/2 ) THEN
         Split total number of streams |c_ij| into two halves
         Route one half from s + N to d on L(B)
         Route another half from s to d on L(A)
      ELSE
         Route traffic from s to d on L(A)
   ELSE IF(d < s) THEN
      IF((s − d) > N/2) THEN
         Route traffic from s to d + N on L(A)
      ELSE IF( (s − d) = N/2 ) THEN
         Split total number of streams |c_ij| into two halves
         Route one half from s to d + N on L(A)
         Route another half from s to d on L(B)
      ELSE
         Route traffic from s to d on L(B)
   c_sd ← c_sd − 1
END WHILE
END ASSIGNMENT
```

Figure 7: Algorithm for assignment of segments on two unidirectional rings to emulate a bidirectional ring.

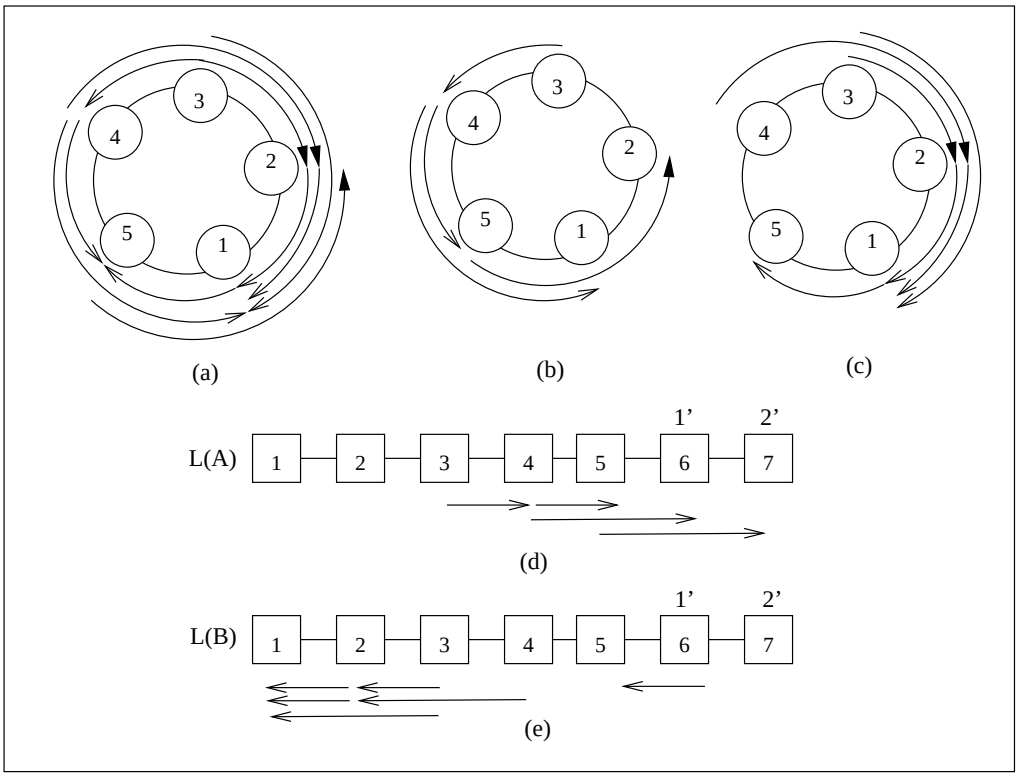

Figure 8: Conversion of a Bidirectional Ring into linear topology. $\lfloor \frac{N}{2} \rfloor$ nodes are added as dummy nodes. In (a) a sample traffic set is shown, while (b) and (c) shows how traffic in (a) can be split between between two unidirectional rings. Mapping of rings in (b) and (c) is shown in (d) and (e), respectively.

Algorithm ASSIGNMENT is self-explanatory. Figure 8, shows how the traffic can be split between two different rings. Also it depicts the assignment of the traffic on corresponding extended linear topologies. The generality of the above mentioned assignment strategy is evident by noting that we are able to handle both symmetric and asymmetric traffic, and are still able to use the same heuristics that we developed for the extended linear topology for unidirectional ring.

IV.1 Non-Shortest Path Routing

In this sub section we will explore non-shortest path routing to reduce the number of the wavelengths and the ADMs. The above mentioned assignment algorithm uses the shortest path to route the traffic between different source destination pairs. However, shortest path may not always lead to the least number of the wavelengths and the ADMs. This can be best illustrated with the help of an example. In Figure 9, two ring A and B are shown. Suppose g in this case is 3. Density d of

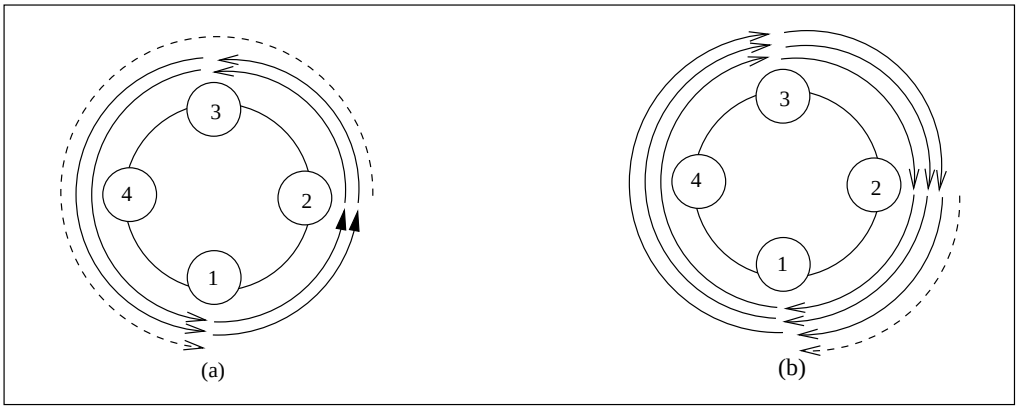

Figure 9: Example to illustrate that the shortest path routing does not necessarily gives the minimum number of the wavelengths and ADMs.

ring A currently is 2 while that of ring B is 3. Currently the number of ADMs required on each ring is also 3. Suppose we need to route a traffic stream originating from node 2 and terminating at node 1. Using shortest path routing we will end up selecting ring B. However, this will not only increment the number of required wavelengths for ring B but also needs two more ADMs. On the other hand, if we choose the longer route on ring A, the already present ADMs will be sufficient to accommodate the stream. This example clearly illustrates that there could arise situations where using shortest path routing doesn't lead to the minimum number of the wavelengths and the ADMs. Hence, to address this issue in the following we will develop an algorithm, TRAFFIC-SHIFTING, that uses three different criteria to relax the shortest path routing restriction, and is given in Figure 10.

In general, we will first assign all the traffic streams on $L(A)$ and $L(B)$ using shortest path. After that we will repeatedly select the ring with larger density and check each of the non-locked segments passing through its maximum density link, selecting the longest segment first. The selected segment is then checked for its eligibility to move to the other ring using CRITERION 1, 2 or 3. In case the shifting of segment is approved, the segment is shifted (now being routed over longer path) and is locked to prohibit any further shifting (to avoid cycles). Note that, to be able to predict exactly that a shifting of the segment from $L(P)$ to $L(Q)$ will lead to the reduced number of the ADMs, we need to determine the corresponding wavelengths of all other segments. Due to our two step approach, however, we do not determine the corresponding wavelengths before routing all of the segments. Hence, at this stage our approach tries to reduces the total number of strings on each ring in a manner that will not lead to an increment in the number of ADMs. The three criteria

Algorithm TRAFFIC-SHIFTING

input:traffic matrices of ring A and B
output:An assignment of traffic streams on $L(A)$ and $L(B)$

BEGIN
Step 1: Assign traffic on both rings using algorithm ASSIGNMENT
Step 2: Compute density $d(A)$ and $d(B)$ for $L(A)$ and $L(B)$, respectively.
Step 3: Determine rings P and A, such that,
  IF $(d(A) \geq d(B))$ THEN $P = A$ and $Q = B$
  ELSE $P = B$ and $Q = A$
Step 4: For each, longest-first, non-locked segment of $L(P)$ that passes through link(s) with density d, DO
  Step 4.1: Apply CRITERION 3, or CRITERION 2, or CRITERION 1 to approve the segment
  Step 4.2: If a segment $c_{ij}^{(k)}$ is approved
    Step 4.2.1: Assign $c_{ij}^{(k)}$ to $L(Q)$ from j to i,
    Step 4.2.2: Lock $c_{ji}^{(k)}$ on $L(Q)$
    Step 4.2.2: Go to step 2
Step 5: Terminate the program
END TRAFFIC-SHIFTING

Figure 10: Algorithm for traffic shifting from shortest path to longer path, to reduce the number of required ADMs.

used in algorithm TRAFFIC-SHIFTING are explained below.

CRITERION 1: Approve the segment if $d(Q)$ is not an integral multiple of g, else try CRITERION 2.

CRITERION 2: Approve the segment if $d(P) > d(Q)$, else try CRITERION 3.

CRITERION 3: Approve the segment if shifting the segment from $L(P)$ to $L(Q)$ does not increase $d(Q)$.

The intuition behind CRITERION 1 is that if $d(Q)$ is not an integral multiple of g then some *space* will be left in at least one of the wavelengths of ring Q, and hence we can utilize it by placing a segment in it, while potentially decreasing the number of strings from the ring P. CRITERION 2 tries to balance the difference of densities (and hence number of strings) between both rings. Finally CRITERION 3, shifts the segment from one ring to another only if such a move does not increase the density of either of the two rings. And hence potentially decreasing the number of strings from both the rings, P and Q, but not at the expense of ring P. Note that using CRITERION 3 fewer segments will be allowed to shift as compared to CRITERION 2 and CRITERION 1. Similarly CRITERION 2 is more restrictive than CRITERION 1.

V Experimental Results

In this section we will present the results of applying the techniques proposed in Sections III and IV to various networks with different topologies and parameters. We are more interested in conducting experiments with nonuniform and asymmetric traffic, because uniform traffic can be considered as a special case of general arbitrary traffic.

We divided the experiments into three suites, corresponding to unidirectional topologies, bidirectional topologies, and the comparison with other similar work reported in the literature, respectively. The following parameters were used for the experiments in suite 1 and 2. The number of nodes N were varied from 5 to 25 with an increment of 5. The value of the grooming factor, g, was assigned to 1, 4, 8, and 16 in each of these experiments. Assuming that our basic data stream is an OC-3, these values of g then correspond to OC-3, OC-12, OC-24, and OC-48, respectively.

For each of the network topologies, between different node pairs (i, j), we generated a set of traffic streams c_{ij} whose cardinality is taken from an integer uniform distribution in the closed interval $[0, g]$. Each reported result is an average value obtained by running 30 batches of 30 runs each. The confidence intervals were computed, but are not shown here.

Experiments in suite 1 were conducted on unidirectional rings. Figure 11 shows the number of the ADMs required for a unidirectional ring, for different values of g. Note that when $g = 1$, there will be no traffic grooming, because each basic stream will occupy the whole wavelength between source and destination nodes. Figure 12 compares the number of the wavelengths determined by MIN-STRINGS for a unidirectional ring, and the corresponding lower bound on the number of the wavelengths, i.e., *density*, when $g = 8$. From Figure 12, it is evident that the number of the wavelengths determined by MIN-STRINGS is exactly equal to lower bound when the problem size is small (for example, when number of nodes are 5 and 10), and slightly exceeds the lower bound when the problem size increases. In Section III, we discussed the time-cost tradeoff in opening the ring at single or multiple nodes. Figure 13 shows the savings in the number of the ADMs that can be obtained by opening the unidirectional ring at each of the N nodes and selecting the best solution. The saving in the number of the ADMs increases when the problem size increases (and so does the computing time). On average we are able to save 5 to 10 ADMs by opening it at each of the N nodes and selecting the best solution. Given that a single port SONET ADM costs \$40,000 or more, this saving could mean a total saving of \$200,000 to \$400,000. On the other hand by opening the ring at each of the N nodes, the time complexity increases by a factor of N. However, the

run time of our technique allows one to afford this additional computation to save large amounts of money while designing WDM networks. For example, the real time taken by the program, for $N = 20$ and $g = 8$, was 0.62 seconds and 5.2 seconds when it was opened at node zero and when it was opened at each of N nodes, respectively. Similarly, the real time taken by the program for the above mentioned two options was 2.71 seconds and 21.7 seconds when $g = 16$. Comparison of our results with other proposed techniques, e.g., [12] reveals that with far less complexity (at least an order of N) we have achieved either less or comparable costs in terms of the number of the ADMs.

For all of the above experiments the amount of the traffic generated between each node pair is related to the grooming factor (the traffic generated for each node pair is an integer uniformly distributed in the closed interval $[0, g]$). We also conducted experiments to study the effect of the different grooming factors while using the same input traffic matrices. For these experiments, the traffic generated for each node pair is an integer uniformly distributed in the closed interval $[0, 16]$. The results are collected for $g = 4$ and $g = 8$. Figures 14 and 15 show the number of the ADMs and the number of the wavelengths required to accommodate the input traffic, respectively. Note that the number of the wavelengths required for $g = 8$ are almost exactly half of that required for $g = 4$, while the number of the ADMs required for $g = 8$ are close to half of that required for $g = 4$. This shows that our two-step approach is scalable with the grooming factor.

In the case of rings even the first step of our two-step technique, i.e., minimizing the number of strings, is NP-Complete. Therefore, to compare the performance of MIN-STRINGS, in this case, we compared the results MIN-STRINGS with the results given in [11]. In [11] the authors solved the wavelength assignment problem in WDM rings while minimizing the number of ADMs. They proposed three heuristics, namely, *Modified Assign First* (MAF), *Iterative Matching* (IMat) and *Iterative Merging* (IMer), and presented the *ADM savings* over 200 experiments. Whenever two segments shared a common node this was counted as an ADM saving. Table 1, compares the performance of MIN-STRINGS to the MAF, IMat, and IMer heuristics, for the same network setup ($N = 16$, and number of streams generated randomly between 16 and 256). On average, our proposed MIN-STRINGS algorithm performed 204%, 145%, and 117% better than MAF, IMat, and IMer, respectively, which is shown by Improvement1 in Table 1. We also experimented by opening the ring at each of the N nodes, and selecting the best solution. This introduced an improvement over our initial solution which is about 8% (shown in Table 1 as Improvement2). This translates into a further improvement of 20%, 16%, and 14%, over MAF, IMat, and IMer, respectively.

Suite 2 consists of experiments for bidirectional rings. Figure 16 shows the number of the ADMs

	MIN-STRINGS	MAF	IMat	IMer
Avg ADM Saving	76	25	31	35
Improvement1	-	204%	145%	117%
Improvement2	8%	224%	161%	131%

Table 1: Average ADM saving for unidirectional rings by MIN-STRINGS versus MAF, IMat, and IMer.

required for a bidirectional ring, using shortest path and TRAFFIC-SHIFTING algorithm, when $g = 16$. Note that the results were collected for the TRAFFIC-SHIFTING algorithm with all three different criteria, namely, CRITERION 1 (C1), CRITERION 2 (C2), and CRITERION 3 (C3). From the figure, it is evident that we can improve on the shortest path approach by relaxing the *shortest path* constraint. Also both C1 and C2 perform better than C3, because they have more flexibility in shifting the traffic streams from shortest path routes to other routes. Table 2 shows the saving in the number of ADMs obtained by using C1, C2, and C3 over the shortest path routing (C0) option, when $g = 8$. Using either C1 or C2, for large problem sizes we were able to save 74 ADMs. On average, criteria C1, C2 and C3 saved 36.1, 35.6, and 6.2 ADMs, respectively, over shortest path option. Using \$40,000 as the price for a single port SONET ADM, this saving translates into 1.44 million, 1.42 million, and 0.24 million dollars, respectively.

N	C0 - C1	C0 - C2	C0 - C3
10	5.7	6	0.5
15	18.4	17.2	1.4
20	46.5	45.3	9.4
25	74	74	13.8

Table 2: Saving in Number of ADMs using TRAFFIC-SHIFTING algorithm for bidirectional ring when $g = 8$.

Suite 3 consists of experiments conducted to compare the similar work done by authors in reference [12]. We selected the same parameters as reported in [12] to be able to make meaningful comparisons. We will report four different experiments. As, Zhang and Qiao provided many results for uniform traffic, in Figure 17 we compared the performance of our algorithms for g=4 and a

uniform traffic of 3 units between randomly selected node pairs, on a unidirectional ring. We used both variations: opening the ring at a single link (*Single open*), and opening the ring at each of the N links and selecting the best results (N *open*). However, in this case both variations yield the same results. We attribute this to the uniform nature of the traffic between different node pairs. Figure 17 shows the number of ADMs saved by using our algorithms over that of reported in reference [12]. Keeping in mind that each ADM costs thousand of dollars the saving in the number of ADMs is substantial. For second experiment we generated a non-uniform traffic in the closed interval [0,5] for a unidirectional ring. Figure 18 shows the results when g=4. We notice that the performance of our algorithms increase over that reported in [12], when non-uniform traffic is considered. Also, opening the ring at each of the N links, though increases the complexity, saves substantial number of ADMs while accommodating the very same traffic. The third experiment is also conducted on a unidirectional ring. Non-uniform traffic is generated in the closed interval [0,5]. However, grooming factors of 8 and 16 are considered. We computed the *Saving Percentage in ADMs*, as defined in reference [12], $S = (N.W - D)/(N.W)$, where as D was defined earlier as the total number of SONET ADMs required to accommodate the traffic. Figure 19 shows that for both g=8 and g=16 in most of the cases the saving percentage of ADMs generated by our algorithms is more than that reported in reference [12]. Finally, the fourth experiment is conducted on a bidirectional ring. Grooming factors of 8 and 16 are considered and a non-uniform traffic is generated in the closed interval [0,5]. Once again results, shown in Figure 20, illustrate that in most of the cases the saving percentage of ADMs obtained by our algorithms is more than that reported in reference [12]. Thus this suite of experiments demonstrates that our proposed algorithms can generate cost-effective solutions in less, or at most equal, computation complexity.

VI Conclusions

In this paper, we address the grooming of the non-uniform traffic on unidirectional and bidirectional rings. We map the unidirectional rings onto a linear topology, and then develop a two-step approach to solve the grooming problem, while minimizing the number of the wavelengths and the ADMs, for the mapped topologies. For the first step, an algorithm MIN-STRINGS is developed that produces the optimal (minimum) number of strings on a linear topology, while compacting each string with traffic streams. Optimality of the algorithm is proven. For the second step, an effective heuristic is designed to group g strings for each wavelength such that the number of the ADMs used per

wavelength are minimized. Also, the bidirectional rings are mapped onto unidirectional rings and the two-step approach is used. Moreover, a study is conducted on routing strategies for bidirectional rings to minimize the number of the required wavelengths and ADMs. Few approaches are proposed that lead to considerable reduction in the number of the required wavelengths and ADMs. Finally, the efficacy of the proposed techniques is demonstrated using a large set of experiments.

References

[1] R. Ramaswami and K. Sivarajan, *Optical Networks: A Practical Perspective*, Morgan Kaufmann, 2nd edition, 2001.

[2] A. L. Chiu and E. H. Modiano, "Traffic Grooming Algorithms for Reducing Electronic Multiplexing Costs in WDM Ring Networks," *Journal of Lightwave technology*, Vol. 18, No. 1, pp. 2-12, January 2000.

[3] E. Modiano, "Traffic Grooming in WDM Networks," *IEEE Communications Magazine*, pp. 124-129, July 2001.

[4] R. Dutta and G. Rouskas, "Traffic Grooming in WDM Networks: Past and Future," *IEEE Network*, Vol. 16, No. 6, pp. 46-56, November/December 2002.

[5] K. Zhu and B. Mukherjee, "A Review of Traffic Grooming in WDM Optical Networks: Architecture and Challenges," *Optical Networks Magazine*, Vol. 4, No. 2, March/April 2003.

[6] O. Gerstel, R. Ramaswami, and G. Sasaki, "Cost-Effective Traffic Grooming in WDM Rings," *IEEE/ACM Transactions on Networking*, Vol. 8, No. 5, pp. 618-630, October 2000.

[7] O. Gerstel, P. Lin, and G. Sasaki, "Combined WDM and SONET Design," In *Proceedings of IEEE INFOCOM'99*, pp. 734-743, 1999.

[8] X. -Y. Li, L. -W. Liu, P. -J. Wan, and O. Frieder, "Practical Traffic Grooming Scheme for Single-Hub SONET/WDM Rings," In *Proceedings of IEEE 25th Annual Conference on Local Computer Networks (LCN)*, Tampa, Florida, pp. 556-564, November 2000.

[9] R. Dutta and G. N. Rouskas, "On Optimal Traffic Grooming in WDM Rings," *IEEE Journal on Selected Areas in Communication*, Vol. 20, No. 1, pp. 110-121, January 2002.

[10] P. -J. Wan, G. Calinescu, L. Liu, and O. Frieder, "Grooming of Arbitrary Traffic in SONET/WDM BLSRs, " *IEEE Journal of Selected Areas in Communications*, pp. 1995-2003, 2000.

[11] L. Liu, X. Li, P. -J. Wan, and O. Frieder, "Wavelength Assignment in WDM Rings to Minimize SONET ADMs, In *Proceedings of IEEE INFOCOM'2000*, pp. 1020-1025, 2000.

[12] X. Zhang and C. Qiao, "An Effective and Comprehensive Approach for Traffic Grooming and Wavelength Assignment in SONET/WDM Rings," *IEEE/ACM Transactions on Networking*, Vol. 8, No. 5, pp. 608-617, October 2000.

[13] W. Cho, J. Wang, and B. Mukherjee, "Improved Approaches for Cost-Effective Traffic Grooming in WDM Ring Networks: Uniform-Traffic Case," *Photonic Network Communications*, Vol. 3, No. 3, pp. 245-254, July 2001.

[14] O. Gerstel, P. Lin, and G. Sasaki, "Wavelength Assignment in a WDM Ring to Minimize Cost of Embedded SONET Rings," In *Proceedings of IEEE INFOCOM'98*, pp. 94-101, 1998.

[15] P.C. Fishburn, *Interval Orders and Interval Graphs*, John Wiley, New York, 1985.

[16] M. Garey, D. Johnson, G. Miller, and C. Papadimitriou, "The complexity of coloring circular arc graphs and chords, " *SIAM Journal on Algebraic and Discrete Methods*, Vol. 1, No. 2, pp. 216-227, 1980.

[17] U. Gupta, D. Lee, and J. -T. Leung, "Efficient Algorithms for Interval Graphs and Circular-Arc Graphs, " *Networks*, Vol. 12, pp. 459-467, 1980.

[18] K. Zhu and B. Mukherjee, "Traffic Grooming in a WDM Mesh Network," *IEEE Journal on Selected Areas in Communications*, Vol. 20, No. 1, pp. 122-133, January 2002.

[19] S. Thiagarajan and A. K. Somani, "Traffic Grooming for Survivable WDM Mesh Networks," In *Proceedings of OPTICOMM 2001*, August 2001.

Appendix

Lemma 1: $|\psi_j| \leq d$ for $1 \leq j \leq |C|$

Proof: $|\psi_j| > d$ implies that at some link, say l, where $l = Xmin(a_j)$, the number of segments traversing the link is greater than d, which contradicts the definition of d. Hence $|\psi_j| \leq d$. ■

Corollary: $|\pi_i| \leq d$ for $1 \leq i \leq N-1$

Proof: By definition, $\pi_i = \psi_m$ for $1 \leq m \leq |C|$, or $\pi_i = \emptyset$. However, from Lemma 1 we have $|\psi_m| \leq d$. Hence, $|\pi_i| \leq d$. ■

Lemma 2: In each iteration, k, of the MIN-STRINGS algorithm (lines 3-18, Figure 4), exactly one member of each nonempty $\pi_i, 1 \leq i \leq N-1$, will be selected for inclusion in string k.

Proof: As MIN-STRINGS first sorts all the segments in ascending order with respect to their $Xmin$ coordinates, in each iteration k (lines 3-18, Figure 4), the sequence Π will be inspected in order, and if the element, π_i, is not empty then:

(a) Either the segment that was chosen in the previous nonempty element, π_j, is not a member of π_i, and therefore does not overlap with the members in π_i. In this case, a new segment in π_i can be chosen for inclusion in the string, and will be removed from set π_i. Or,

(b) the segment, a_k, that was last chosen from the previous non-empty member, π_j, is also a member of π_i, and will also be removed from π_i.

In both cases, the size of all the nonempty sets, π_i, which are members of the sequence L, will be reduced by 1. ■

Theorem 1: MIN-STRINGS algorithm is optimal in the number of strings.

Proof: As d is the lower bound on the number of strings, we will prove that the number of strings obtained by MIN-STRINGS algorithm is equal to d, i.e., $|R| = d$, and is therefore optimal. Since $|\pi_i| \leq d, 1 \leq i \leq (N-1)$, then by lemma 2, in d or less iterations we will be able to select all the segments $a_k \in \pi_i$, for the strings. Let k^* be the link with density d. Then $|\pi_{k^*}| = d$. Hence the total number of iterations required to select all the segments is d. Since each iteration corresponds to a string, then $|R| = d$. ■

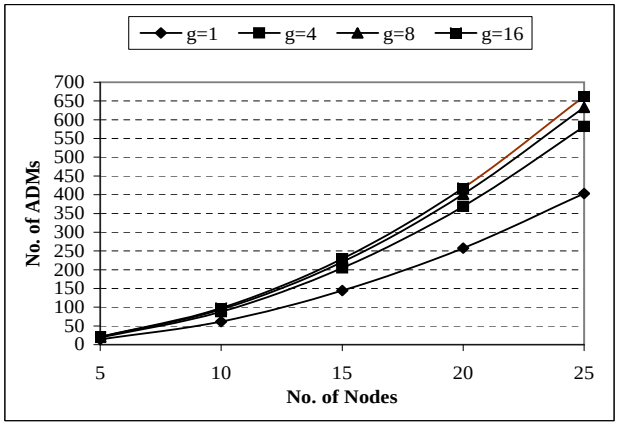

Figure 11: Number of the ADMs for a unidirectional ring

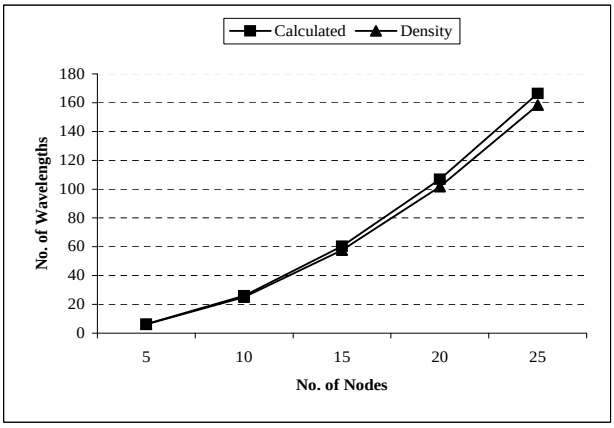

Figure 12: Number of the wavelengths and the corresponding lower bound for a unidirectional ring

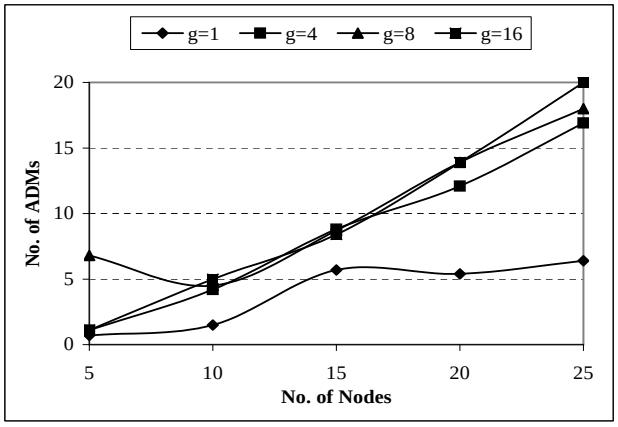

Figure 13: Number of the ADMs saved by opening the unidirectional ring at each of the N nodes

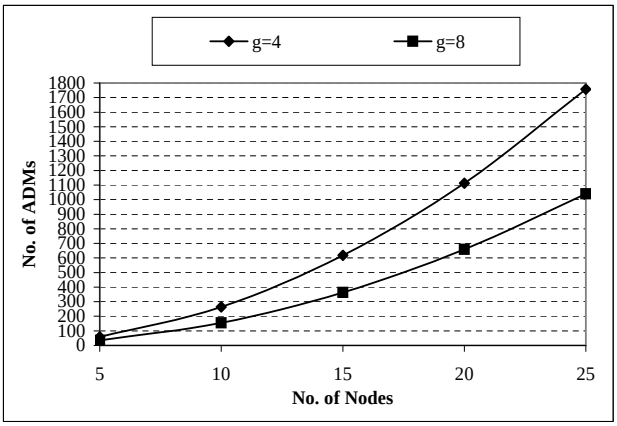

Figure 14: Number of the ADMs when the same input matrices are used for the experiments with $g = 4$ and $g = 8$

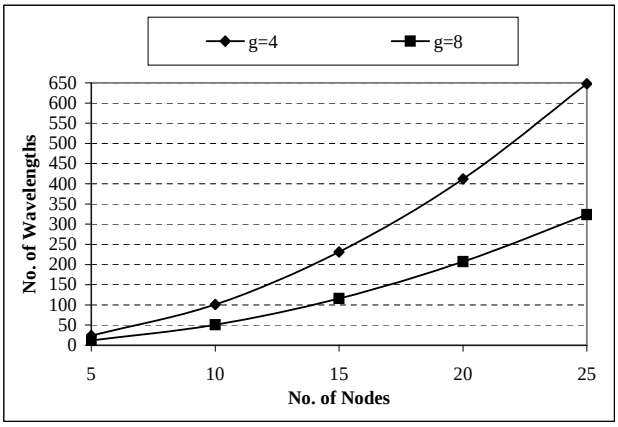

Figure 15: Number of the wavelengths when the same input matrices are used for the experiments with $g = 4$ and $g = 8$

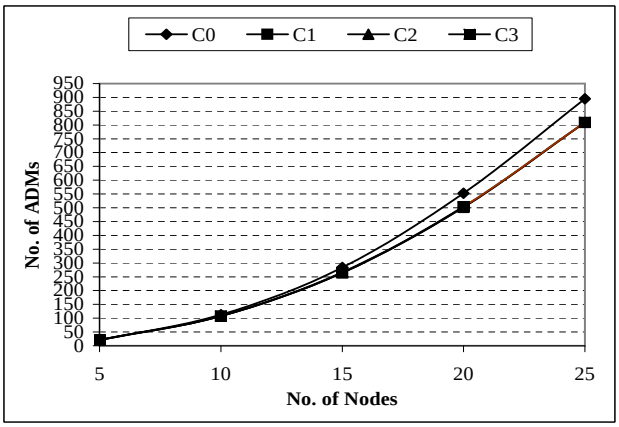

Figure 16: No. of the ADMs for a bidirectional ring; $g = 16$

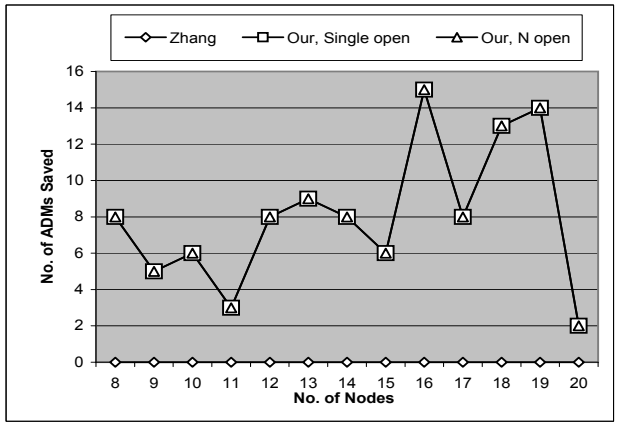

Figure 17: Number of the ADMs saved with uniform traffic and $g = 4$ on a unidirectional ring

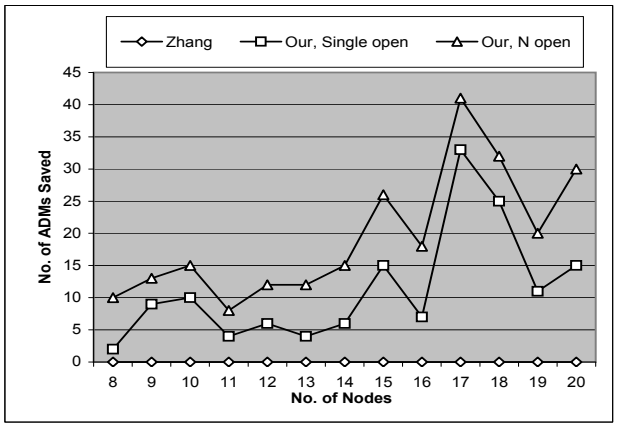

Figure 18: Number of the ADMs saved with nonuniform traffic and $g = 4$ on a unidirectional ring

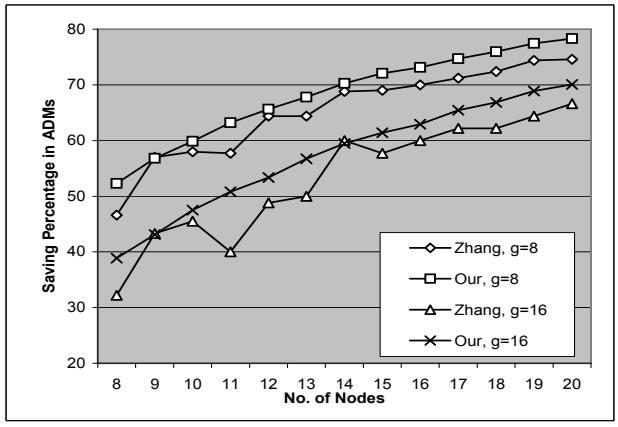

Figure 19: Saving percentage in ADMs, with nonuniform traffic, $g = 8$ and $g = 16$, on a unidirectional ring

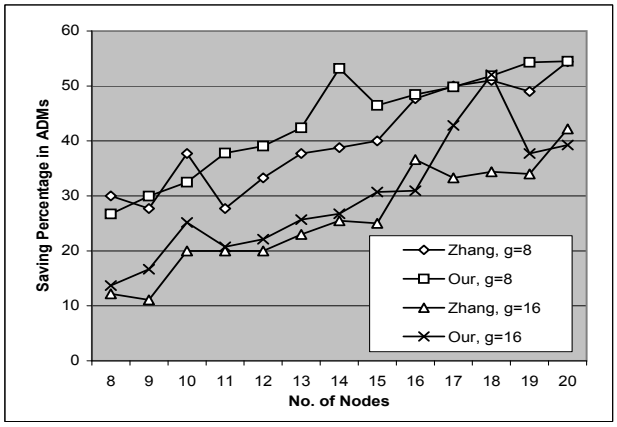

Figure 20: Saving percentage in ADMs, with nonuniform traffic, $g = 8$ and $g = 16$, on a bidirectional ring