

Traffic grooming in WDM SONET UPSR rings with multiple line speeds

Huan Liu, Fouad A. Tobagi
 Department of Electrical Engineering, Stanford University
 huanliu, tobagi@stanford.edu

Abstract—We consider the traffic grooming problem in WDM/SONET UPSR rings with multiple line speeds. This is motivated by the fact that when traffic demands are non-uniform and are spread over a relatively wide range, a ring using multiple line speeds would lead to a lower cost, owing to the economy of scale seen in devices, in particular, electronic ADMs, running at higher speeds. We give a novel Integer Linear Programming (ILP) formulation for the problem. We also propose two techniques to exploit the symmetric problem structure resulting from the equivalency in the line speed assignments and the color assignments for wavelengths. The techniques reduce computation time, thus, allow many problem instances to be solved exactly. For large size problems, we propose efficient heuristic algorithms that achieve a similar cost using a fraction of the computation time. We show that, by allowing WDM/SONET rings to run at different line speeds, we can greatly reduce the ADM cost. We also show that allowing nodes to switch traffic can help to reduce the cost further.

I. INTRODUCTION

WDM/SONET architecture is gaining popularity because it provides an easy upgrade path, avoiding the need to lay out additional fibers to meet the traffic growth. In WDM/SONET, many rings run in parallel and each ring runs on a separate wavelength. Each node needs one Optical Add-Drop Multiplexer (OADM), which allows the node to add/drop a selected subset of the wavelengths into and out of the fiber. Each node may also need up to W electronic ADMs, one for each wavelength on which the node originates or terminates some traffic demands.

A traffic demand can be routed on any one of the wavelengths. If the traffic demand is routed on one wavelength, then an ADM has to be installed on that wavelength at both the source and destination nodes. Fig. 1 shows an example of the node architecture. In the example, a fiber with three wavelengths is connected to the node through an OADM. Wavelength w_2 and w_3 are dropped at the node, and the other wavelength (w_1) is passed through transparently. Thus, only two electronic ADMs are needed at the node. The ADMs can add or drop individual traffic streams into the corresponding wavelength. An optional Digital Cross Connect (DCC) may be also present at the node to switch traffic from one ADM to another.

A major design goal of a WDM/SONET network is to minimize the total equipment cost by intelligently arranging the traffic demands onto the different SONET rings. This is commonly referred to as traffic grooming. Since an OADM is needed at every node, the cost of OADMs is fixed. However,

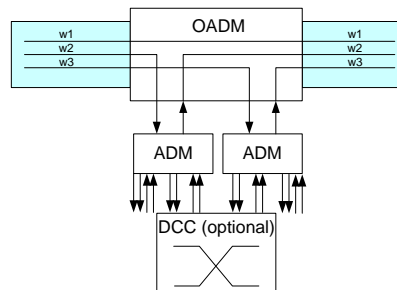


Fig. 1. Node architecture

the cost of electronic ADMs varies depending on how traffic is groomed. A naive solution is to install an ADM on every wavelength at each node. Unfortunately, such a solution is very costly. An example is given in [1] which shows that significant cost savings could be achieved by intelligently arranging traffic demands onto different rings. The potential savings can be significant and grow with both the network size and internodal demand [2]. Although only very simple traffic demand matrices are considered in [2], we would expect the savings to be significant for other traffic patterns too.

Most prior work on traffic grooming assumes that all SONET rings run at the same line speed. Such a restriction could greatly increase the cost especially when the traffic demands are non-uniform. In this paper, we consider the traffic grooming problem where each SONET ring can be assigned to different line speeds and study the potential cost savings by using multiple line speeds.

There are two types of SONET rings: Unidirectional Path Switched Rings (UPSR) and Bidirectional Line Switched Rings (BLSR)[3]. In UPSR, traffic is always routed in one direction (e.g., clockwise). To route a symmetric traffic stream, i.e., the same amount of traffic from node A to B and from node B to A, the same amount of capacity is reserved on every fiber along the ring. Therefore, no spatial capacity reuse is possible. UPSR ring consists of 2 fibers where one of the fibers is dedicated for protection purpose. In BLSR, traffic can be either routed in the clockwise or routed in the counter-clockwise direction. BLSR consists of either two or four fibers. They are commonly referred to as BLSR/2 and BLSR/4 respectively. UPSR is commonly used in Metro access networks, whereas BLSR is commonly used in Metro backbone networks. In this paper, we restrict ourselves to

UPSR only. We note that similar results can be obtained for BLSR as well.

The traffic grooming problem where all SONET rings are restricted to the same line speed was first considered in [4] [5] [3]. It was first considered in two separate steps: traffic routing [4] and wavelength assignment [5]. In the subsequent paper [3], it was shown that considering the traffic grooming problem in one single step could lead to lower cost. Traffic grooming in SONET UPSR rings with one single line speed has been considered in [6], where heuristic algorithms were proposed for specific traffic patterns, and in [7] [8], where an Integer Linear Programming (ILP) formulation was given. These results do not apply when a wavelength can choose from a set of available line speeds. Traffic grooming in BLSR or bidirectional SONET rings with only one line speed has been studied in [9] [10] [11] [12] [13]. Traffic grooming in mesh topologies has been studied in [14][15].

Prior work on traffic grooming with multiple line speeds can be found in [3] [16] [17]. Traffic grooming in UPSR rings with two line speeds was first briefly discussed in [3]. The authors compared network cost using a simple lower bound. The lower bound indicates that having two line speeds available might lower network cost under certain cases. Traffic grooming in UPSR rings with a central hub node was considered in [16] [17] where all traffic terminates at the central hub node. The authors did not take the number of wavelengths as a constraint, as a result, the solution may not be feasible if only a limited number of wavelengths are available. No problem formulation or general solution methodology has been given in [3] [16] [17], and furthermore, their analyses assume specific cost and capacity ratios between two different line speeds. The result cannot be easily generalized when the ratios are different.

In this paper, we study the cost benefit of using multiple line speeds. To solve the traffic grooming problem, we investigate both the ILP approach which can solve the problem optimally and the heuristic approach which can solve a much larger problem instance at the cost of less than optimal results. Our solution approaches can be applied to both the single line speed and the multiple line speeds cases. Similar to prior work on traffic grooming [3] [16] [17] [6] [7], we limit ourselves to consider only bidirectional traffic demands. However, we make no specific assumption regarding the cost and speed ratios between different line speeds. Unlike previous work [17] [16] where no limit on the number of wavelengths is assumed, we take the number of wavelengths available as a constraint, since a WDM system typically supports a fixed number of wavelengths, such as 8, 12 or 40 wavelengths, in a single fiber.

The paper is organized as follows. In section II, we investigate the ILP approach which can solve the problem optimally. We present a novel ILP formulation and two techniques that exploit the symmetric problem structure. The novel formulation and the techniques greatly reduce the runtime, thus, allow us to solve much larger problem instances and allow us to further validate our heuristic algorithms. In addition, the new ILP formulation allows us to study the cost benefit

of allowing traffic to switch across rings at some nodes. In section III, we present efficient heuristic algorithms that can obtain comparable results in a much shorter amount of time. In section IV, we evaluate our algorithms and approaches. Then, in section V, we present some numerical results. Finally, in section VI, we conclude the paper.

NOMENCLATURE

- N : The number of nodes in the ring, i.e., the number of traffic add/drop sites around the ring. Nodes are labeled $0, 1, \dots, N - 1$ in the clockwise direction.
- W : The number of wavelengths available in the WDM system.
- R : The number of line speeds available. For example, if OC-3, OC-12 and OC-48 line speeds are available, $R = 3$.
- K : Total number of commodities in the aggregate formulation.
- w : When used as subscript, it denotes a wavelength. $w \in 1, \dots, W$.
- r : When used as subscript, it denotes a line speed. $r \in 1, \dots, R$.
- i, j : They denote nodes on the rings. $i, j \in 1, \dots, N$.
- k : It denotes a commodity. $k \in 1, \dots, K$.
- $s(k)$: The source node of the k th commodity.
- $d(k)$: The destination node of the k th commodity.
- $f(i, j)$: $f(i, j)$ is an integer that describes the total traffic demand from node i to node j in terms of the lowest traffic granularity. For example, if the lowest traffic granularity is OC-3, and the traffic demand between node i and j is one OC-48 circuit, then $f(i, j) = 16$. We assume $f(i, i)$ is always 0.
- g_r : Capacity of the r th line speed in multiples of the lowest traffic granularity. If the lowest traffic granularity is OC-3, then line speed at OC-3 has capacity of 1, line speed at OC-12 has capacity of 4 and line speed at OC-48 has capacity of 16. We assume that the capacity is an integer number throughout this paper.
- c_r : The cost of an ADM running at the r th line speed. An OC-48 ADM will cost more than an OC-12 ADM, which in turn cost more than an OC-3 ADM.
- M : A large constant.

II. ILP APPROACH

The traffic grooming problem can be formulated as an Integer Linear Programming (ILP) problem and solved using commercial ILP solvers. The ILP formulation for the single line speed case was first given in [7] [8]. Two variables are defined in this formulation: y_{iw} which is 1 if an ADM needs to be installed at node i on wavelength w , and x_{kw} which denotes how much traffic of the k th demand is routed on wavelength w . Since our objective is to minimize the total ADM cost and all ADMs in the single line speed case cost the same, the optimization objective will thus be minimizing the sum of all y_{iw} variables.

Several constraints make sure that the solution will be feasible. The first constraint, called the demand met constraint, makes sure that every traffic demand is routed on at least one ring. The second constraint, called the capacity constraint, makes sure that the total capacity of each ring is not violated. Lastly, an additional constraint, called the ADM constraint, relates y_{iw} to x_{kw} variables. It forces the y_{iw} variable to be 1 if an x_{kw} variable is nonzero and node i is either the source or the destination node of the k th traffic demand.

The ILP for the single line speed case can be easily extended to the multiple line speeds case [18] by creating R different rings for each wavelength, each running at one of the R available line speeds. Since there are R rings for each wavelength, the y_{iw} variables become y_{iwr} , and the x_{kw} variables become x_{kwr} . Then, an additional constraint is added to make sure that only one of the R rings has nonzero capacity. We use the term “ring wr ” to refer to the ring running at the r th line speed on wavelength w .

The ILP for the multiple line speed case [18] is shown as follows. We use δ_{wr} variables to denote the line speed of a wavelength. It is 1 if ring wr can carry traffic, otherwise it is 0.

$$\text{(MILP1)} \quad \min \sum_{i=1}^N \sum_{w=1}^W \sum_{r=1}^R c_r y_{iwr} \quad (1)$$

Subject to

$$\sum_{r=1}^R \sum_{w=1}^W x_{kwr} = f(s(k), d(k)) \quad \forall k \quad (2)$$

$$\sum_k x_{kwr} \leq \delta_{wr} g_r \quad \forall w, r \quad (3)$$

$$M y_{iwr} \geq \sum_{k|i=s(k)} x_{kwr} + \sum_{k|i=d(k)} x_{kwr} \quad \forall i, w, r \quad (4)$$

$$\sum_{r=1}^R \delta_{wr} \leq 1 \quad \forall w \quad (5)$$

$$x_{kwr} \text{ are integers} \quad (6)$$

$$\delta_{wr}, y_{iwr} \in \{0, 1\} \quad (7)$$

Equation (2) is the demand met constraint. Equation (3) is the capacity constraint. Equation (4) is the ADM constraint. Equation (5) ensures that only one out of the R rings on wavelength w can carry traffic.

The simple extension unfortunately complicates the formulation. As a result, only very small problem instances can be solved. For example, within the same amount of computation time, a network with 9 nodes and 10 wavelengths can be solved in the single line speed case. In the 3 line speeds case, however, we are only able to solve for a network with 4 nodes and 10 wavelengths.

In this section, we propose several techniques that can be used to reduce the computational complexity, and thus allow a much larger problem to be solved.

Our first technique reduces computation time by using a more efficient alternative formulation. Such a formulation not only cuts down solution time, but also allows us to study the cost benefit of allowing traffic to switch across rings.

The other two techniques exploit the fact that there are a large number of equivalent solutions in the solution space as a result of the line speed assignments and the color assignments. The second technique eliminates the duplication in line speed assignments. The third technique eliminates the duplication in color assignments for wavelengths having the same line speeds. Both techniques allow us to search only the unique solution space, therefore cut down the solution space dramatically. The reason that there are a large number of equivalent solutions is because, in the demand met constraint of the ILP (e.g. equation (2)), we only make sure that the sum of traffic routed on all rings meets the traffic demand, but we are not specifying which set of rings the traffic should be routed on. Therefore, given a solution to the traffic grooming problem, we could permute the traffic assignment from one ring to another and obtain an equivalent solution.

These proposed techniques can be used together to obtain combined computation time reduction. With these techniques, we are able to solve a problem instance in a much shorter amount of time. If we have to terminate the computation early for large size problems, the techniques allow us to explore a larger portion of the solution space in a fixed amount of time, therefore, we are able to obtain better solutions.

It is worth noting that other techniques have also been proposed to reduce the computation time. For example, in [8], the authors attempt to reduce the computation time by exploiting the block diagonal structure in the constraint matrix and use column generation technique to decompose the problem into smaller and easier to solve sub-problems. The column generation technique can be used to speed up the computation of the relaxed Linear Programming problem at any iteration of a branch and bound algorithm.

A. Aggregate problem formulation

The ILP formulation [7][8][18] can be considered as a variant of the multi-commodity flow problem. It needs more computation time because each traffic demand is treated as a separate commodity. In this section, we present a much more scalable formulation, called the aggregate formulation. The aggregate formulation treats all traffic demands originating from the same source node (or terminating at the same destination node) as one single commodity. In the aggregate formulation, the number of commodities is always less than the number of nodes ($K \leq N$). Compared to the number of traffic demands which can be as high as $N(N-1)/2$, it is a very large improvement. As a result, the aggregate formulation could take much less computation time. Aggregate formulation has been used in network design problems [20] and in Logical Topology design problems [21] where it showed promise for reduction in computational complexity.

In the ILP formulation, instead of using a single variable x_{kwr} to represent a traffic demand, we can use two variables: x_{kiwr}^+ at the source node and x_{kjwr}^- at the destination node. The benefit of such a transformation is that we can now define several traffic demands as a single commodity, thus reduce the computational complexity. In this paper, we treat all traffic

demands that originate from the same node as one single commodity. Then we only need to use one x_{kiwr}^+ variable at the source node, but we will need a separate x_{kiwr}^- variable at each of the destination nodes of this commodity.

Because we are using two variables for a traffic demand now, we need to introduce an additional flow conservation constraint which makes sure that the same amount of traffic originating from some nodes on a ring will terminate at some other nodes on the same ring. In addition, the rest of the constraints need to be modified to be expressed in terms of the new variables.

With a single variable (x_{kwr}), each traffic demand has to stay on the same ring on its path from its source node to its destination node. But if we can use Digital Cross Connects (DCC) to switch traffic from one ring to another at a particular node, the total ADM cost could be reduced further. With two variables, the aggregate formulation can take traffic switching into account and allow us to study the cost benefit of traffic switching. To switch some traffic f of the k th commodity from ring wr to $w'r'$ at node i , we can set $x_{kiwr}^- = f$, which says that f amount of traffic arrives at node i on ring wr , and we can then set $x_{kiw'r'}^+ = f$, which says that f amount of traffic leaves from node i on ring $w'r'$.

In general, the more nodes we allow switching, the higher the potential for ADM cost savings. Fig. 2 shows an example where allowing node 2 and 5 to switch traffic can result in lower total ADM cost. In the figure, a pair of squares are used to denote the source and destination node of a traffic demand. The link between them shows the path the traffic demand follows. The traffic demand between node 1 and 3 does not need a separate ADM at node 3 on ring 3, it can switch to ring 2 at node 2 and then terminate at the ADM at node 3 on that ring. Similarly, for traffic demand between node 4 and 6, we can save one ADM by switching traffic at node 5.

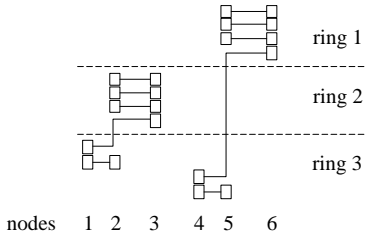


Fig. 2. Allowing two nodes to switch traffic results in lower total ADM cost

We use the following variables in the aggregate formulation.

- x_{kiwr}^+ : It is the total amount of traffic for commodity k that leaves from node i on ring wr . The traffic could either originate from node i , or could be switched from another ring at node i if traffic switching is allowed.
- x_{kiwr}^- : It is the total amount of traffic for commodity k that arrives at node i on ring wr . The traffic could either terminate at node i , or could be switched to another ring at node i if traffic switching is allowed.
- y_{iwr} : It is a binary variable indicating whether traffic will

be added or dropped at node i on ring wr . If it is 1, an ADM running at the r th line speed has to be installed at node i on wavelength w .

- δ_{wr} : It is a binary variable. It is 1 if ring wr can carry traffic, otherwise it is 0.

In the following, we first present the aggregate formulation without traffic switching, then we show the formulation that takes traffic switching into account.

1) *Problem formulation with no traffic switching*: The formulation with no traffic switching capability is shown below.

$$(MILP2) \quad \min \sum_{i=1}^N \sum_{w=1}^W \sum_{r=1}^R c_r y_{iwr} \quad (8)$$

Subject to

$$\left\{ \begin{array}{l} \sum_{r=1}^R \sum_{w=1}^W x_{kiwr}^+ = \\ \quad \sum_{j=1}^N f(i, j) \quad \text{if } s(k) = i \\ \sum_{r=1}^R \sum_{w=1}^W x_{kiwr}^- = \\ \quad f(s(k), i) \quad \text{otherwise} \end{array} \right. \quad \forall k, i \quad (9)$$

$$\sum_{i=1}^N (x_{kiwr}^+ - x_{kiwr}^-) = 0 \quad \forall w, r, k \quad (10)$$

$$\sum_{k=1}^K \sum_{i=1}^N x_{kiwr}^+ \leq \delta_{wr} g_r \quad \forall w, r \quad (11)$$

$$M y_{iwr} \geq \sum_{k=1}^K (x_{kiwr}^+ + x_{kiwr}^-) \quad \forall i, w, r \quad (12)$$

$$\sum_{r=1}^R \delta_{wr} \leq 1 \quad \forall w \quad (13)$$

$$x_{kiwr}^+, x_{kiwr}^- \text{ are integers} \quad (14)$$

$$\delta_{wr}, y_{iwr} \in \{0, 1\} \quad (15)$$

The equations above are explained as follows.

- The minimization objective (8) is to minimize the total ADM cost.
- Equation (9) is the demand met constraint. It ensures that the total traffic demand for each commodity is met. Note that we treat all traffic demands originating from the same source node as a single commodity.
- Equation (10) is the flow conservation constraint. It states that any traffic originates on a ring has to terminate on the same ring for any commodity k .
- Equation (11) is the capacity constraint. It makes sure that the total traffic flow on any ring is less than its capacity. We only need to count flow that originates on the ring, hence, we are only adding x_{kiwr}^+ variables.
- Equation (12) is the ADM constraint. It forces y_{iwr} variables to be 1 if any traffic is added or dropped at node i on ring wr .
- Equation (13) ensures that only one out of the R rings on wavelength w can carry traffic.

The solution time for a mixed integer linear programming problem is heavily dependent on the number of integer

variables in the formulation, because a branch and bound algorithm has to branch on each integer variable in turn to arrive at the optimal solution. The following theorem shows that the actual number of integer variables in the formulation is much smaller than it appears. The proof can be found in [19].

Theorem 1: If the integer constraint on x_{kiwr}^+ and x_{kiwr}^- variables (equation (14)) is removed in the MILP2 formulation, there is still an optimal solution where the x_{kiwr}^+ and x_{kiwr}^- variables are integers.

The proof for the theorem is constructive. If after removing the integer constraint (equation (14)), the ILP solver finds an optimal solution to the MILP2 formulation that has fractional x_{kiwr}^+ and x_{kiwr}^- variables, we can easily construct another optimal solution where all x_{kiwr}^+ and x_{kiwr}^- variables are integers.

Even if the integer constraint on x_{kiwr}^+ and x_{kiwr}^- variables can be removed, the large number of these variables can still greatly slow down the computation time. It may seem that there are $KNWR$ number of x_{kiwr}^+ variables and an equal number of x_{kiwr}^- variables. Fortunately, a large number of these variables are 0, therefore, they do not have to be included in the formulation. The x_{kiwr}^+ variables are non-zero only when $s(k) = i$, therefore, only KWR of these variables are actually used in the formulation. Similarly, if $f(s(k), i)$ is zero (i.e. no traffic from commodity k terminates at node i), the corresponding x_{kiwr}^- variables will be zero, therefore, they can be left out of the formulation. As a result, there are only $K'WR$ number of x_{kiwr}^- variables used in the formulation, where K' is the total number of traffic demands.

Note that this formulation solves exactly the same problem as that of MILP1. Although this formulation has both more variables and more constraints than MILP1, as hinted at by earlier work in a different area [21] [20] and confirmed by our experimental study, this formulation actually greatly reduce the computation time. On average, the new formulation takes half as much time. The reason that this new formulation performs better is that it exposes the problem structure better, as a result, it helps the branch and bound algorithm to find the optimal integer solution faster.

2) *Problem formulation with traffic switching:* The formulation given in the last subsection only allows traffic flow to stay on the same ring from its source node to its destination node. If we allow traffic to be switched at any node from one ring to another, potentially large savings are possible. There are two ways of switching traffic at a node as described in [3]. One is by manually connecting one traffic port from an ADM to one traffic port on another ADM sitting on a different wavelength. Another is to utilize DCC which can switch traffic streams through programming control.

To allow traffic switching at every node, we only need to replace equation (9) in the MILP2 formulation by the following equation. The equation makes sure that the total traffic demand for each commodity is met. Because all nodes can now act as switching nodes, a pair of x_{kiwr}^+ and x_{kiwr}^- variables have to be introduced for each node. There are

$KNWR$ number of x_{kiwr}^+ variables and an equal number of x_{kiwr}^- variables.

$$\sum_{r=1}^R \sum_{w=1}^W (x_{kiwr}^+ - x_{kiwr}^-) = \begin{cases} \sum_{j=1}^N f(i, j) & \text{if } s(k) = i \\ -f(s(k), i) & \text{otherwise} \end{cases} \quad \forall k, i \quad (16)$$

It is also possible to allow only certain nodes to switch traffic by controlling what variables are used. We introduce both x_{kiwr}^+ and x_{kiwr}^- variables for any node i that is capable of switching traffic, and apply equation (16) to make sure that the traffic demands are met. For other nodes, we apply equation (9), and only introduce x_{kiwr}^+ and x_{kiwr}^- variables if they are non-zero.

Note that, even if only one node is capable of switching traffic, it is still different from the hub architecture as described in [4][6][16][22]. In a hub architecture, all traffic has to traverse to the hub before it is forwarded to its final destination. Whereas in our formulation, we only switch traffic at the node if it helps to reduce the network cost. Some traffic may bypass the hub (switching) nodes if routing it directly can reduce cost.

3) *Aggregate commodities:* The aggregate formulation treats multiple traffic demands as one single commodity. Since our traffic model is bidirectional, it is equivalent to express a traffic demand from node i to j as a traffic demand from node j to i . Given the choices, we should choose how to express the traffic demands intelligently in order to reduce the total number of commodities. Since the complexity of the formulation grows as the number of commodities increases, reducing the number of commodities will help to keep the problem formulation small, allowing it to be solved quickly. In the following, we show how the run time can be further reduced by changing how a traffic demand is expressed.

The problem of determining how each traffic demand should be expressed such that the total number of commodities is minimized can be easily transformed into a vertex covering problem. The vertex covering problem is NP-complete. Although many heuristic algorithms exist, we propose to solve it exactly because our problem is small enough (SONET can have at most 16 nodes).

The problem can be formulated as the following ILP problem.

$$\min \sum_{i=1}^N x_i \quad (17)$$

$$s.t. \quad x_{s(k)} + x_{d(k)} \geq 1 \quad \forall k \quad (18)$$

$$x_i \in \{0, 1\} \quad (19)$$

We use the x_i variables to indicate whether a commodity will be defined for node i . If so, all traffic demands that have node i as one of their end nodes should be expressed as originating from node i , so that they all become part of the commodity. Each traffic demand corresponds to one constraint as expressed by equation (18). The constraint makes sure that at least one commodity is defined at either of the two end

nodes of the traffic demand. If a commodity is defined at both end nodes, i.e., $x_{s(k)}$ and $x_{d(k)}$ are both 1, then the traffic demand can be expressed as originating from either node. The objective function (17) tries to minimize the total number of commodities.

One advantage of the ILP approach is that we can easily extend the MILP formulation to accommodate a number of other constraints and considerations. For example, the formulation could take switching cost into consideration, take limited number of ADM equipment as a constraint or disallow bifurcation of traffic. The detail of these can be found in [19].

B. Exploit symmetry in line speed assignments

The ILP formulation contains a large number of duplicate solutions. Consider a solution to the traffic grooming problem where the i th wavelength is set to the r_i th line speed and the j th wavelength is set to the r_j th line speed, we can construct an equivalent solution by setting the i th wavelength to use the r_j th line speed and setting the j th wavelength to use the r_i th line speed, then moving all traffic routed on wavelength j to wavelength i and moving all traffic routed on wavelength i to wavelength j . The solution thus constructed is clearly identical to the original solution, however, the ILP formulation treats them as separate solutions.

In [18], we gave a non-linear problem formulation. Instead of assuming that there are R rings for each wavelength, we assume that there is only one ring. Then in the capacity constraint, we change the capacity of the ring based on which line speed is chosen. This unfortunately makes the objective function non-linear. To solve it, we proposed to decompose the problem into many subproblems, where each subproblem corresponds to one possible way of line speed assignment. Because the line speed assignment is determined already, the subproblems' objective function becomes linear again. The decomposition not only allows us to solve each subproblem using an ILP solver, but more importantly, it also allows us to eliminate duplicate solutions by choosing to solve only the subproblems that correspond to unique line speed assignments.

C. Exploit symmetry in color assignments

The technique presented in the last section only eliminates duplicate solutions resulting from the line speed assignments to wavelengths. There are still a large number of duplicate solutions resulting from the equivalency in the color assignments. In this section, we propose to exploit it through a new multi-variable branch and bound technique.

Consider a solution to the traffic grooming problem and any two wavelengths running at the same line speed: w and w' , we can easily permute w and w' to obtain another equivalent solution. Again, the ILP formulation treats them as separate solutions.

The number of duplicate solutions resulting from the equivalency is large. Consider the y_{iw} variables in the ILP formulation for a single line speed. Each y_{iw} variable can take on the value of either 0 or 1. Therefore, the total number of all combinations of value assignments is $2^{N \times W}$. However, the

number of unique combinations after eliminating equivalent solutions is much smaller. To derive the formula for the number of unique combinations, let us consider one node at a time. Without loss of generality, let us assume the node is 0. There are 2^W combinations of value assignments for the y_{0w} variables. However, the number of unique combinations is small. We note that the number of y_{0w} variables that are assigned 1 determines whether a combination is unique. Therefore, there are only $W + 1$ unique combinations. To determine the total number of unique combinations, we can enumerate all possibilities recursively. Let $C_{N,W}$ denote the number of unique combinations with N nodes and W wavelengths, then $C_{N,W}$ can be recursively defined as follows:

$$C_{N,W} = \sum_{w=0}^W C_{N-1,w} \times C_{N-1,W-w}$$

Note that $C_{N,W}$ is 1 if $N = 0$ or $W = 0$.

The number of all combinations is much larger than the number of unique combinations. As an example, with 20 wavelengths, the number of all combinations is more than 10^{18} times larger than that of the unique combinations. The difference grows as the number of wavelengths increases. This is expected because the number of duplicated solutions grows exponentially as the number of wavelengths.

We should note that a branch and bound routine will not necessarily explore all combinations because it can frequently prune large solution space based on the current bound. Therefore, the extra time taken for a branch and bound routine to explore the whole solution space rather than only the unique solution space may not be proportionally longer. Still, we expect significant savings if we could limit our search only to the unique solution space.

The branch and bound algorithm normally branches on one fractional integer variable at a time. This works well on many problems. However, it performs poorly on problems with a symmetric structure. This is because even though we force a variable with fractional value to be integer when we branch, the same fractional value will re-appear on other variables because of the symmetry. For example, if a y_{iw} variable is fractional and we force it to be either 1 or 0, then another $y_{iw'}$ variable will become fractional because wavelength w and w' are symmetric and it is equivalent to install an ADM on w or on w' . To force an integer solution, all $y_{iw} \forall w$ variables have to be forced to integer values eventually through the branch and bound algorithm. The number of branch nodes created in this process is large and it is proportional to 2^W .

To overcome this problem, we propose a new multi-variable branch and bound technique which will branch on multiple variables at a time instead of only on one single variable. For example, if a y_{iw} variable is fractional, we will branch on all $y_{iw} \forall w$ variables at the same time. Because of the symmetric structure, we do not have to create 2^W number of branch nodes. Instead, we only need to create $W + 1$ nodes, one for each unique value assignment.

When some integer variables have been fixed to some particular values at any step of the branch and bound algorithm,

the symmetric structure will change. For example, if y_{iw} is fixed to 1 and $y_{iw'}$ is fixed to 0, i.e., an ADM is installed at node i on wavelength w , but not on wavelength w' , then $y_{i'w}$ and $y_{i'w'}$ variables are no longer equivalent to each other. Our branch and bound algorithm keeps track of what variables have been fixed in order to determine which set of variables to branch on to eliminate the symmetry.

The amount of extra information to keep track of is minimal. Assuming we have v number of variables, and assuming that we use depth first policy in the branch and bound algorithm, then there can at most be v number of branch and bound nodes at any time. For each branch and bound node, we only need to keep $O(W)$ amount of information on variable equivalency. Therefore, we need $O(vW)$ amount of extra storage, which is quite a small amount of overhead.

Because of the length limitation, we refer interested readers to a technical report [19] for the full details of this technique.

III. HEURISTIC ALGORITHMS

The ILP formulation can be solved exactly to get the optimal solution. Although we can now solve a reasonable size problem optimally with our techniques, it is still time consuming to solve large size problems. For these large problems, we propose efficient heuristic algorithms. Our improvements on the ILP approach allow us to extensively validate our proposed heuristic algorithms, and show that the heuristic algorithms can give near optimal results within a much shorter amount of time.

We note that our algorithms are the first heuristic algorithms to handle arbitrary traffic demand matrices, even in the case where only one line speed is available.

Our heuristic algorithms route traffic onto one ring at a time. When routing traffic onto a ring, we try to minimize the average cost per unit of routed traffic. Once some traffic has been routed on one ring, we proceed to the next ring to route the remaining traffic. Our algorithms are greedy in nature since we route traffic onto a ring based on the best average cost on that ring. We note that such a local optimal decision may not necessarily lead to the global optimal.

In the following, we first describe the CPD ratio which is the essential criteria we use to determine which traffic to route onto each ring. Then we will describe the heuristic algorithms.

A. The CPD ratio

Our heuristic algorithms are based on the idea of reducing the *Cost Per unit Demand* (CPD) ratio. The CPD ratio has been used in [6][11] to derive a lower bound and in [23] to derive optimal solutions for uniform all-to-all traffic demands. However, our algorithms are the first to use the CPD ratio to guide the search for a solution. We first define the CPD ratio in the following.

Given a traffic matrix T , we want to route some traffic demands on a ring running at the r th line speed. For an arbitrary integer n ($n \leq N$), let $D(n, r)$ denote the maximum amount of traffic among any n nodes in T that could be routed on the ring running at the r th line speed, and

let $T(n, r) = [t(n, r)_{sd}]$ denote the corresponding traffic matrix that is routed on the ring. By definition, we have $D(n, r) = \sum_{sd} t(n, r)_{sd}$. Furthermore, because there is no spatial capacity reuse in UPSR, and because a ring running at the r th line speed can support at most g_r traffic streams, $g_r \geq D(n, r)$. The CPD ratio $\rho(n, r)$ is then defined as the ratio between the ADM cost (n ADMs each costing c_r) and the amount of routed traffic ($D(n, r)$).

$$\rho(n, r) = \frac{c_r \times n}{D(n, r)}$$

The CPD ratio is essentially the minimum average cost to route some traffic streams on a ring running at the r th line speed using n nodes. Naturally, the lower the ratio, the lower the cost to route $D(n, r)$ traffic streams. Note that the CPD ratio is defined for a specific traffic matrix T . If the traffic matrix is different, the corresponding CPD ratios will be all different.

In general, there could be many traffic matrices $T(n, r)$ such that $D(n, r) = \sum_{sd} t(n, r)_{sd}$. Any of these traffic matrices could be routed on the ring to achieve the same CPD ratio. In our heuristic algorithms, we arbitrarily pick one of the traffic matrices.

We say a CPD ratio $\rho(n, r)$ *dominates* another CPD ratio $\rho(n', r')$ if $\rho(n, r) \leq \rho(n', r')$ and $D(n, r) \geq D(n', r')$. If no other CPD ratios dominate $\rho(n, r)$, we say $\rho(n, r)$ is a *dominant* CPD ratio, and we say $\rho(n, r)$ is a *non-dominant* CPD ratio if otherwise.

Our heuristic algorithms are greedy in nature. Therefore, there is no reason we should use a non-dominant CPD ratio to route traffic demands. For a non-dominant CPD ratio $\rho(n, r)$, we can always find another lower CPD ratio $\rho(n', r') < \rho(n, r)$ such that it can accommodate more traffic streams on the same ring, i.e., $D(n', r') \geq D(n, r)$. When we route traffic demands onto a ring, we first evaluate the CPD ratio $\rho(n, r)$ for all possible n and all possible r , and remove all non-dominant CPD ratios. We then sort the remaining dominant CPD ratios in increasing order, and number them based on their order as follows:

$$\rho(n_1, r_1) \leq \rho(n_2, r_2) \leq \rho(n_3, r_3) \leq \dots$$

Because we have removed all non-dominant CPD ratios, the corresponding amount of routed traffic should be in sorted order as well, i.e.,

$$D(n_1, r_1) \leq D(n_2, r_2) \leq D(n_3, r_3) \leq \dots$$

It is desirable to use $\rho(n_1, r_1)$ when routing traffic demands onto each ring because it has the lowest average cost. Unfortunately, this may not be possible because we only have a limited number of wavelengths available. Thus we need the higher CPD ratios $\rho(n_i, r_i)$ where $i > 1$ to pack more traffic onto a ring so that we can route all traffic demands on the W wavelengths.

$\rho(n, r)$ can be easily computed by examining all $\binom{N}{n}$ combinations. To compute all CPD ratios at a particular line

speed, we have to examine $\sum_{n=2}^N \binom{N}{n} < 2^N$ combinations. This may appear to be time consuming at first glance. Fortunately, in SONET architecture, N is limited to at most 16, and examining all 2^{16} combinations only takes a very small amount of time.

B. The HCPDF heuristic algorithm

In general, we want to use the lowest dominant CPD ratio as much as possible, and only use the higher dominant ratios as necessary to pack more traffic onto a ring so that we can fit all traffic demands into the W wavelengths. When we are forced to use a higher dominant CPD ratio, it is beneficial to use it early on, i.e., use the higher CPD ratio first on ring 1, then on ring 2, and so on. This is because when we are routing traffic on the first few rings, there is more unrouted traffic, so that there are more chances for a more optimized solution. Our first heuristic tries to use the higher CPD ratios early, therefore, we call it the High CPD ratio First (HCPDF) algorithm.

We maintain a set of pointers p_i , one for each ring. They are used to remember which CPD ratios should be used on a particular ring, i.e., on the i th ring, the p_i th dominant ratio should be used.

The HCPDF heuristic is shown in the following algorithm.

Algorithm HCPDF_heuristic

Input: T , N and W

Output: Cost of routing traffic demands T using W wavelengths

- 1 Initialize
 - 1.1 Set $p_i = 1, \forall i$
 - 2 Route traffic demands based on the current p_i .
 - 2.1 For each ring i from 1 to W , repeat step 2.2
 - 2.2 Find the p_i th dominant CPD ratio, then route $T(n_{p_i}, r_{p_i})$ on the i th ring. Set $T = T - T(n_{p_i}, r_{p_i})$, then repeat step 2.2 for other rings.
 - 3 Check if feasible
 - 3.1 If all traffic demands are routed in step 2, a solution has been found, return
 - 3.2 Otherwise, find the largest i such that $p_{i-1} > p_i$, and increment p_i . If all p_i are equal, increment p_1 . Then for all $j > i$, reset $p_j = 1$. Return to step 2.

Initially, all p_i are 1, meaning that we will use the best CPD ratio for each ring. In the second step, we route all traffic demands based on the set of pointers p_i . We first compute all dominant CPD ratios for the first ring, then we choose the p_1 th CPD ratio. In other words, we route $T(n_{p_1}, r_{p_1})$ traffic on the first ring. We repeat the process for other rings as well, i.e., we use the p_i th dominant CPD ratio when routing traffic on

the i th ring. This process continues until all traffic demands are routed, or no more rings are left.

In the third step, we first determine if a feasible solution has been found. If all traffic demands have been routed, we return. Otherwise if there is still traffic to route, but there are no more rings left, we will pick a ring i and increase p_i by 1 so that we can route more traffic on that ring. Because more traffic is routed on ring i , there is a chance that we might be able to route the remaining traffic using lower CPD ratios. Therefore, we reset all $p_j = 1, \forall j > i$.

We pick a p_i to increase by 1 such that the relationship of $p_1 \geq p_2 \geq p_3 \dots$ is always maintained. We do so by choosing the largest i such that $p_{i-1} > p_i$. If all p_i are equal, we choose the first pointer p_1 to increase by 1. The reason we maintain this relationship is because if we need to use a higher CPD ratio to route more traffic streams, we should do so as early as possible when there are more unrouted traffic, so that there are more chances for a more optimized solution.

Note that if a pointer p_i is changed in step 3.2, then the CPD ratios have to be recalculated in step 2.2 in the next iteration for any wavelength j such that $j > i$. This is because when a different CPD ratio is used on wavelength i , a different set of traffic is routed on wavelength i . Since the remaining traffic for the rest of the wavelengths will be different, the CPD ratios will be different.

We also propose another heuristic algorithm, called the All CPD ratio (ACPD) algorithm, which more exhaustively search for the right CPD ratio to use at each ring. The ACPD algorithm gives slightly better results than HCPDF, but uses more computation time. The details of the ACPD algorithm can be found in [19].

IV. ALGORITHMS AND TECHNIQUES EVALUATION

To evaluate our proposed formulation and techniques, we consider three SONET networks supporting OC-3 traffic streams. All traffic demands are expressed as multiples of OC-3 circuits. The first network has only one line speed, e.g., OC-48. This is the network considered in [4] [6]. The second network has two line speeds available, e.g., OC-12 and OC-48. This is the network considered in [3] [17]. Lastly, we also consider a network with three line speeds, i.e., OC-3, OC-12 and OC-48. Even though we use OC-3, OC-12 and OC-48 in our examples, the same principle applies to higher line speeds such as OC-192 and OC-768.

We consider several traffic patterns for evaluation, including the uniform traffic pattern with 1 OC-3 demand between every node pair, the central traffic pattern with 1 OC-3 demand between every node and a central hub node and the random traffic pattern where the demands are randomly generated.

We use the same cost ratio assumption as in [3], i.e., OC-48 ADMs are 2.5 times more costly than OC-12 ADMs, and OC-12 ADMs are 2.5 time more costly than OC-3 ADMs. The capacity ratio is 4, i.e., OC-48 has 4 times the capacity of OC-12 and OC-12 has 4 times the capacity of OC-3. We normalize the cost of an OC-3 ADM to be 1. Therefore, an OC-12 ADM will cost 2.5 and an OC-48 ADM will cost 6.25. Even though

we use these set of fixed ratios in our experiments, we note that our formulations and heuristic algorithms are general enough to handle any cost and capacity ratio.

We implemented an ILP solver that incorporates the techniques we proposed, and we also implemented the heuristic algorithms. We run all experiments on a Sun Blade 2000 workstation with UltraSPARC III+ 900Mhz CPUs. All computation time shown is in seconds in CPU time on the Sun workstations. We impose an upper limit on the computation time for the ILP solver to be 10000 seconds. If the ILP solver takes more than the time allowed, we terminate the computation and record the best solution it is able to find up to that point.

We only include limited set of computation comparisons. A full set of comparisons are available in [19].

A. Computational feasibility of the ILP approach

With our proposed techniques, we are able to solve a large set of problem instances optimally. For larger size problems, we can terminate the ILP solver early and use the best solution it can find up to that point. We find that the ILP solver can typically get optimal or near optimal solution very early on, but spends a lot of time searching the whole solution space for better solutions. In Fig. 3, we show the computation progress of the ILP solver for a 5 node 2 line speeds (OC-12 and OC-48) network with 10 wavelengths supporting uniform traffic demands. The top line shows the best integer solution it has found so far, and the bottom line shows the best bound it has found through linear relaxation. These are shown against the number of branch and bound nodes that have been explored (x axis). The solver finds an initial solution with cost of 33, and quickly improves it to the optimal solution of 25. This all happens within the first 100 branch and bound nodes, so it is hard to see in the figure. Even though the optimal solution has been found very early on, the solver still takes a lot of time to improve the bound to confirm the optimality of the current solution. We believe this is a very common behavior for the traffic grooming problem. Therefore, by terminating the computation early, we are able to solve a even larger set of problem instances optimally using the ILP approach.

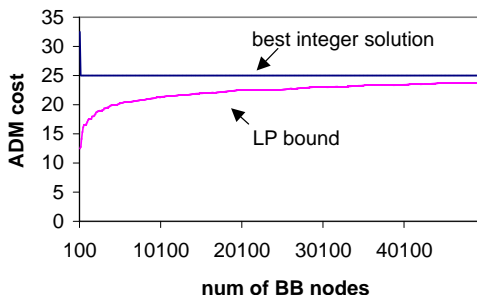


Fig. 3. Computation progress of the ILP solver. Uniform traffic, $W=10$, 5 nodes, 2 line speeds.

To see what problems can be solved optimally using the ILP approach, we consider uniform traffic pattern and a network with 10 wavelengths. Table I lists the size of the network in terms of the number of nodes that can be solved.

TABLE I
THE SIZE OF THE NETWORK THAT CAN BE SOLVED. UNIFORM TRAFFIC,
 $W=10$.

	1 LS	3 LS	3 LS new
Solved exactly	7	4	9
Terminated early but still optimal	13	8	14

Under the column “1 LS”, we show what problems can be solved using the ILP formulation in [7][8] for the single line speed case. Under the column “3 LS”, we show what problems can be solved with 3 line speeds using the simple extended ILP formulation in [18]. Since using multiple line speeds complicates the formulation, we can only solve a much smaller problem. For problems that can be solved exactly, we can only solve it for a network with 4 nodes instead of 7 nodes. For problems where we terminate the computation early but still get the optimal solutions, we can solve for a network with only 8 nodes instead of 13. Under the column “3 LS new”, we show the result of the new formulation and techniques. We now can solve for a network with 9 nodes exactly, and obtain optimal results for a network with 14 nodes if we terminate the computation early.

For central traffic pattern, a network with 10 wavelengths, 3 line speeds and up to 16 nodes can be solved exactly within 200 seconds of computation time. It is worth noting that the central traffic pattern is exactly the traffic pattern in the single hub architecture as considered in [4] [6] [16] [17]. It is remarkable that we can solve these problems exactly within few hundred seconds.

More detailed comparison of the aggregate formulation and the proposed techniques, including their individual contribution to the run time reduction, can be found in [19].

B. Heuristic algorithms evaluation

To evaluate the heuristic algorithms, we consider the random traffic pattern. In our experiments, we randomly generate one traffic matrix for a network with n nodes. The traffic matrix has $\max\{n(n-1)/8, n-1\}$ traffic demands and the size of each traffic demand is randomly chosen between 1 and 2 OC-3 circuits. Random traffic pattern is a good approximation of real life traffic pattern.

The results from the ILP solver and the heuristic algorithms assuming 10 wavelengths and 3 line speeds are shown in Table II. We put an asterisk next to the ILP results when the ILP solver finishes the computation before the allowed time, so those results are known to be optimal. As shown, the HCPDF algorithm can produce solutions that are very close to those from the ILP solver. In one case ($n = 15$), the HCPDF algorithm even produced better result. Also, the HCPDF algorithm takes much less computation time. Even for the largest problem ($n = 16$), it takes only 10 seconds.

TABLE II
HEURISTIC ALGORITHM EVALUATION. RANDOM TRAFFIC, $W=10$, 3 LINE SPEEDS.

# of nodes	ILP cost	HCPDF	
		cost	time
4	7.5*	8	0.04
5	15*	16	0.14
6	16.5*	18	0.21
7	18.5*	20	0.3
8	19.5*	20	0.37
9	23.5*	23.5	0.78
10	34*	34	1.68
11	36.5	37	2.31
12	51	51	3.16
13	57.5	59.5	3.99
14	77.5	83.25	8.68
15	95	85.75	9.43
16	97.5	103.25	10.7

V. NUMERICAL RESULTS

In this section, we use the ILP formulation and the heuristic algorithm to study the cost benefit of traffic switching and the cost benefit of using multiple line speeds.

A. Traffic switching

Traffic switching can lower the network cost as shown in Fig. 2. The aggregate formulation allows traffic to switch across different rings at a selected subset of nodes. Thus, using our novel formulation, we are able to study the cost benefit of traffic switching. We consider a network with only one line speed (OC-48) and 10 wavelengths available. The cost savings for uniform traffic when one node is allowed to switch traffic, when two nodes are allowed to switch traffic and when all nodes are allowed to switch traffic are shown in Fig. 4. For comparison, we also show the case when no traffic switching is allowed. The results are obtained using the ILP solver. As shown, the cost savings when one node can switch traffic are up to 33%. Having more nodes switching only reduces the cost further marginally. It suggests that just setting up switching at one node may be good enough. This observation is consistent with that in [22], where they found that the number of switching nodes should be approximately the same as the number of wavelengths of traffic generated by each node.

For random traffic patterns, we see similar cost savings. However, for central traffic pattern, we are not able to see any cost savings when we introduce switching nodes. This is because traffic switching only helps to reduce cost when an ADM has to be installed at the switching node for other traffic demands anyway. This is not true for central traffic pattern.

B. Multiple line speeds

Multiple line speeds can lower cost in two different cases. The first case happens when no lower line speed is available.

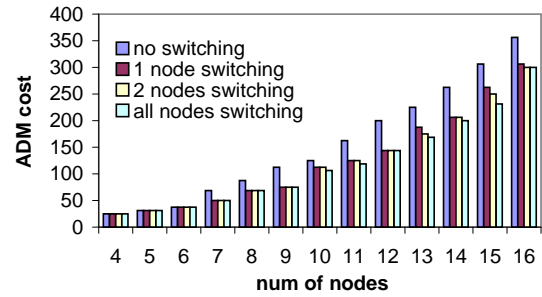


Fig. 4. Cost benefit of traffic switching. Uniform traffic, one line speed, $W=10$

Therefore, one is forced to use the more expensive ADMs even if the traffic to support is small. If lower and less costly line speed is available, we can use that instead to lower the overall cost. For example, if the only line speed available is OC-48 and the traffic demand between two nodes is only one OC-3 circuit, then we can use OC-3 ADMs to lower the cost and still support the same traffic.

The second case happens when no higher line speed is available. Therefore, one can not achieve enough aggregation to realize the economy of scale. For example, if the only line speed available is OC-12 and the traffic demand between two nodes is one OC-48 worth of traffic, then we have to use 8 OC-12 ADMs. But, if OC-48 line speed is also available, we can use only 2 OC-48 ADMs instead. In general, 2 OC-48 ADMs will cost less than 8 OC-12 ADMs.

We now try to characterize the conditions under which multiple line speeds can lower cost. To make the experiments statistically significant, we run each data point 10 times and then take the average, i.e., for each data point used in this section, we randomly generated 10 different instances and then took the average of their cost. We used the HCPDF algorithm for all experiments because it can produce very good results using very little computation time. For these experiments, we specified a large W in order to guarantee that a feasible solution can be found. This will also result in the lowest cost possible with multiple line speeds, because, otherwise, we may be forced to pack traffic using higher cost ADMs.

We first consider a 16 nodes network. The traffic demands are randomly generated, where 1/3 of them require one OC-3 circuit, 1/3 of them require one OC-12 circuit and the remaining 1/3 require one OC-48 circuit. In Fig. 5, we plot the cost savings of using 3 line speeds (OC-3, OC-12 and OC-48) versus using a fixed line speed (OC-3 or OC-12 or OC-48). If only OC-3 line speed is available, the network cost is very high because many ADMs are needed to support one OC-48 traffic. If only OC-12 line speed is available, the network cost is lower because some traffic aggregation can be achieved. If only OC-48 line speed is available, the network cost is high when the number of traffic demands is small because an OC-48 ADM may have to be installed to accommodate an OC-3 traffic. When the number of traffic demands increases, the network cost decreases because it becomes possible to aggregate more OC-3 and OC-12 traffic.

As shown in the figure, no matter which single line speed one chooses, the network cost will always be significantly higher compared to the case where multiple line speeds are used.

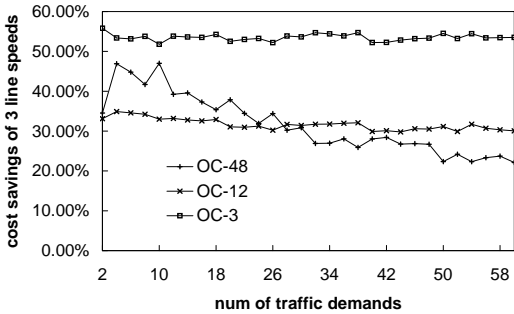


Fig. 5. Cost savings. 16 nodes, 1/3 demands of size OC-3, OC-12 and OC-48

We next consider a 16 nodes network with 16 randomly generated traffic demands. The size of each traffic demand is randomly generated between 1 and a fixed number S , therefore, the average size of a demand is $S/2$. The cost savings of using 3 line speeds as compared to using 1 single line speed are shown in Fig. 6. When S is small, OC-3 costs the least; and when S is big, OC-48 costs the least. When S is in between, using only OC-12 line speed gives the lowest network cost. Again, no matter which fixed line speed is chosen, there is a large region of S where roughly 20% cost savings can be achieved by using 3 line speeds. If the fixed line speed is not chosen intelligently, much higher cost savings could be achieved by using multiple line speeds.

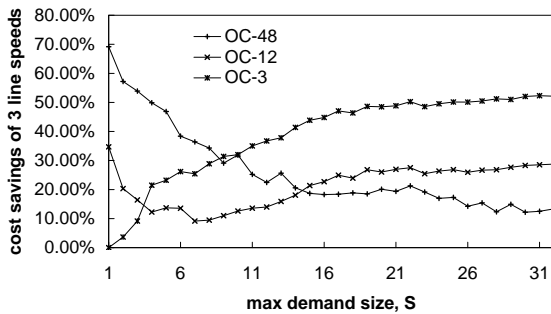


Fig. 6. Cost savings. 16 nodes, 16 demands, each demand ranges from 1 to S

We also show the number of wavelengths used in Fig. 7. As expected, OC-48 uses the least number of wavelengths and OC-12 tends to use many more wavelengths especially when S is large. In comparison, multiple line speeds only uses slightly more wavelengths than OC-48. Since many wavelengths (at least 40) are available in a typical DWDM system, this seems to be a small price to pay for the large reduction in the network cost.

When S is small (< 10), multiple line speeds use more wavelengths than OC-12. This shows that the cost savings are a result of the first case, where introducing lower line speed (OC-3) can lower cost. When S is large (> 10), multiple line

speeds use fewer wavelengths than OC-12. This shows that the cost savings are a result of the second case, where introducing higher line speed (OC-48) can lower cost.

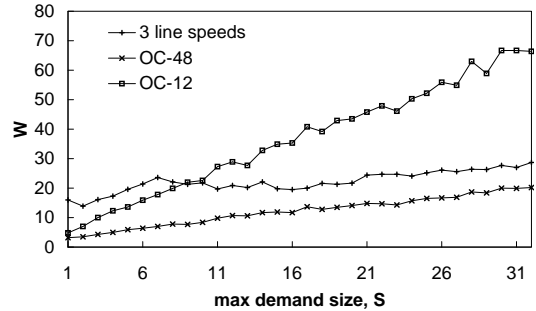


Fig. 7. Number of wavelengths used. 16 nodes, 16 demands, each demand ranges from 1 to S

Multiple line speeds can also increase capacity utilization. The capacity utilization is defined as the ratio between the total amount of traffic routed and the sum of capacity of each wavelength. As shown in Fig. 8, multiple line speeds has much high utilization ratio than OC-48 because aggregation is not always possible. Although multiple line speeds has slightly worst utilization ratio than OC-12 when S is large, we note that OC-12 costs significant more in the same region (Fig. 6).

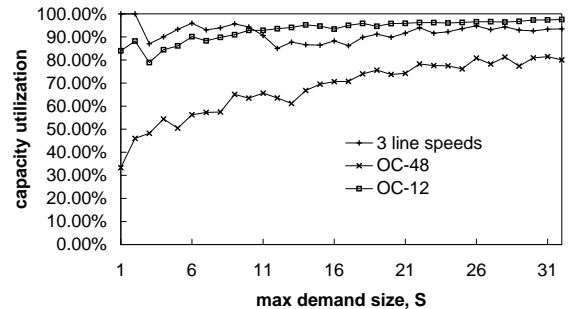


Fig. 8. Utilization ratio. 16 nodes, 16 demands, each demand ranges from 1 to S

The results we have shown (Fig. 5-8) assume that a large number of wavelengths are available. When the number of wavelengths available is small, we are forced to use the more costly ADMs at a higher line speed to route all traffic demands. Therefore, the resulting network cost will be higher. We study the cost savings as a function of the number of wavelengths for a 5 nodes network with 8 randomly generated traffic demands ranging from 1 to 4 OC-3 circuits. The result obtained from the ILP solver is shown in Fig. 9. The total ADM cost decreases as more wavelengths become available. However, no further cost savings could be achieved beyond a certain point. For 2 line speeds, there are no cost savings beyond 5 wavelengths, and for 3 line speeds, there are no cost savings beyond 8 wavelengths. We also note that most of the cost savings could be achieved using only 2 line speeds. Adding the third line speed only reduces the cost further marginally.

So far we have assumed that the cost ratio is exactly 2.5.

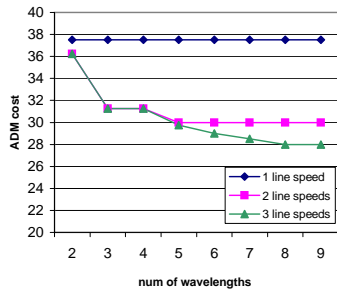


Fig. 9. Reducing ADM cost using more wavelengths. 5 nodes, random traffic, 8 demands, demand size ranges from 1 to 4 OC-3 circuits.

However, in practice, the ratio could vary a lot. We now look at how the cost ratio can affect the savings. We consider a 10 nodes network with 10 traffic demands, each demand ranges from 1 to 16 in size. We vary the cost ratio from 2 to 3 and show the result in Fig. 10.

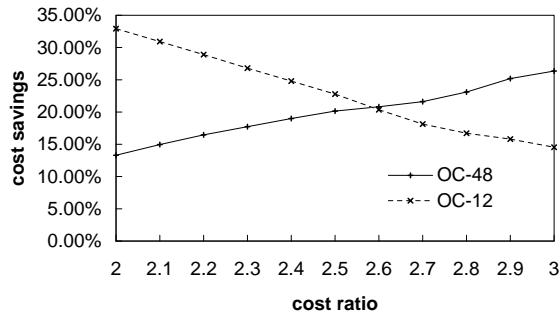


Fig. 10. Cost savings as a function of the cost ratio. 10 nodes, 10 demands.

As cost ratio increases, OC-48 cost increases. This is because multiple line speed has an increasing chance finding lower cost alternatives using lower line speeds. However, as cost ratio increases, OC-12 cost decreases. This is because the cost savings as a result of aggregation tend to diminish as the cost ratio increases.

VI. CONCLUSION

We consider the traffic grooming problem in WDM/SONET UPSR rings with multiple line speeds. We study two different approaches to solve the traffic grooming problem. To solve the problem optimally, we propose a new aggregate ILP formulation and two new techniques that can reduce the computation time, thereby, allowing many problem instances to be solved exactly. For larger problem instances, we propose efficient heuristic algorithms that can derive comparable solutions within a much shorter amount of time.

We show that having more line speeds could greatly reduce the network cost, especially when the traffic demands are non-uniform. We quantify the region where the cost savings are most pronounce. We also show that multiple line speeds can increase capacity utilization. Our novel aggregate formulation allows traffic to switch across rings. Using the formulation, we show that traffic switching can reduce cost further and

that most of the cost savings could be achieved by allowing only one node to switch traffic.

REFERENCES

- [1] E. Modiano and P. J. Lin, "Traffic grooming in WDM networks," *IEEE Commun. Mag.*, pp. 124–129, July 2001.
- [2] J. M. Simmons, E. L. Goldstein, and A. Saleh, "Quantifying the benefit of wavelength add-drop in WDM rings with distance-independent and dependent traffic," *J. Lightwave Technol.*, vol. 17, no. 1, pp. 48–57, Jan. 1999.
- [3] O. Gerstel, P. Lin, and G. Sasaki, "Combined WDM and SONET network design," in *Proc. INFOCOM*, 1999.
- [4] O. Gerstel, R. Ramaswami, and G. H. Sasaki, "Cost-effective traffic grooming in WDM rings," *IEEE/ACM Trans. Networking*, vol. 8, no. 5, pp. 618–630, Oct. 2000.
- [5] O. O. Gerstel, P. Lin, and G. H. Sasaki, "Wavelength assignment in a WDM ring: To minimize cost of embedded SONET rings," in *Proc. INFOCOM*, 1998, pp. 94–101.
- [6] A. L. Chiu and E. H. Modiano, "Traffic grooming algorithms for reducing electronic multiplexing costs in WDM ring networks," *J. Lightwave Technol.*, vol. 18, no. 1, pp. 2–12, Jan. 2000.
- [7] J.-Q. Hu, "Traffic grooming in wavelength-division-multiplexing ring networks: a linear programming solution," *J. of Optical Networking*, vol. 1, no. 11, pp. 397–408, Nov. 2002.
- [8] A. Sutter, F. Vanderbeck, and L. Wolsey, "Optimal placement of add/drop multiplexers: Heuristic and exact algorithms," *Operations Research*, vol. 46, pp. 719–728, 1998.
- [9] J. Wang, W. Cho, V. R. Vemuri, and B. Mukherjee, "Improved approaches for cost-effective traffic grooming in WDM ring networks: ILP formulations and single-hop and multihop connections," *J. Lightwave Technol.*, vol. 19, no. 11, pp. 1645–1653, Nov. 2001.
- [10] T. Y. Chow and P. J. Lin, "The ring grooming problem," *Discrete Applied Math*, vol. 0, no. 0, pp. 1–12, Jan. 2000.
- [11] P.-J. Wan, G. Galinescu, L. Liu, and O. Frieder, "Grooming of arbitrary traffic in SONET/WDM blrs," *IEEE J. Select. Areas Commun.*, vol. 18, no. 10, pp. 1995–2003, Oct. 2000.
- [12] X. Zhang and C. Qiao, "An effective and comprehensive approach for traffic grooming and wavelength assignment in SONET/WDM rings," *IEEE/ACM Trans. Networking*, vol. 8, no. 5, pp. 608–617, Oct. 2000.
- [13] R. Dutta and G. N. Rouskas, "On optimal traffic grooming in WDM rings," *IEEE J. Select. Areas Commun.*, vol. 20, no. 1, pp. 110–121, Jan. 2002.
- [14] K. Zhu and B. Mukherjee, "Traffic grooming in an optical WDM mesh network," *IEEE J. Select. Areas Commun.*, vol. 20, no. 1, pp. 122–133, Jan. 2002.
- [15] J.-Q. Hu and B. Leida, "Traffic grooming, routing and wavelength assignment in optical WDM mesh networks," in *Proc. IEEE INFOCOM*, Mar. 2004.
- [16] X. Y. Li, L. Liu, P. J. Wan, and O. Frieder, "Practical traffic grooming scheme for single-hub SONET/WDM rings," in *Proc. 25th IEEE Conf. on Local Computer Networks*, Nov. 2000.
- [17] X. Li, P. Wan, and L. Liu, "Select line speeds for single-hub SONET/WDM ring networks," in *Proc. ICC*, 2000, pp. 495–499.
- [18] H. Liu and F. Tobagi, "A novel efficient technique for traffic grooming in WDM/SONET with multiple line speeds," in *Proc. ICC*, 2004.
- [19] —, "Traffic grooming in WDM SONET rings supporting multiple line speeds," Stanford university, Tech. Rep., 2004.
- [20] D. Bienstock and O. Gunluk, "Computational experience with a difficult mixed-integer multi-commodity flow problem," *Mathemat. Program.*, vol. 68, pp. 213–237, 1995.
- [21] R. M. Krishnaswamy and K. N. Sivarajan, "Design of logical topologies: A linear formulation for wavelength-routed optical networks with no wavelength changers," *IEEE/ACM Trans. Networking*, vol. 9, no. 2, Apr. 2001.
- [22] R. Berry and E. Modiano, "The role of switching in reducing the number of electronic ports in WDM networks," *IEEE J. Select. Areas Commun.*, vol. 22, no. 8, pp. 1396–1405, Oct. 2004.
- [23] J.-Q. Hu, "Optimal traffic grooming for wavelength-division-multiplexing rings with all-to-all uniform traffic," *J. of Optical Networking*, vol. 1, no. 1, pp. 32–42, Jan. 2002.