Spare Capacity Provisioning for Dynamic Traffic Grooming in Optical Networks

Shu Huang and Rudra Dutta

Department of Computer Science, North Carolina State University, Raleigh, NC 27695, USA
shuang5@unity.ncsu.edu, dutta@csc.ncsu.edu

Abstract—Optical Networks with wavelength routing are expected to form the backbone in the next generation wide area networks. The traffic grooming problem in optical networks is an optimization problem of aggregating sub-wavelength traffic demands onto lightpaths such that the required electronic switching capability, hence network cost, can be minimized. In this paper, we propose an over-provisioning scheme, which pre-allocates the spare capacity of lightpaths to dynamic subwavelength traffic demands such that the network can be more agile in responding to traffic increment requests. For the single-link case, the problem is a convex non-linear programming problem. We provide an exact algorithm to find the optimal solution. The problem with general topologies is then studied. We prove the NP-hardness in this case, and propose heuristics. Numerical results show our heuristics perform well.

I. INTRODUCTION

Wavelength-routed optical networks have long been recognized as the backbone networks of tomorrow. The attractive qualities of optical transmission, together with the technology of routing individual wavelength channels without the need for intermediate Opto-Electro-Optic (OEO) interconversion, has further made such technology attractive for a future in which the backbone will be characterized by the need for high speeds with highly predictable performance. The literature on the *virtual topology design* problem has focused on the possibility of forming *lightpaths* or clear optical channels that can then be viewed as traffic that must be routed and assigned wavelengths on the physical topology of the network [1].

In the course of evolution of next generation networks and networking applications, it has become clear that realistically end-to-end traffic demands are typically considerably smaller than the capacity of whole wavelength channels, and such *sub-wavelength* traffic must be multiplexed (using electronic TDM methods) into individual wavelength channels to obtain good utilization of network bandwidth. This process has been called *traffic grooming*, and the literature has seen a significant amount of interest in this area in recent years [2].

Network design with sub-wavelength traffic demands has grown more and more worthy of consideration because of several reasons. Optical networks are being extended closer and closer to end users, where more flexibilities to set up and tear down low speed traffic commands are required. In the framework of GMPLS, for instance, signaling and routing protocols used in MPLS networks have already been extended to other networks including time-division, spatial

and wavelength switching networks. LSPs (label switched paths) in lower hierarchy are allowed to be tunnelled through LSPs in higher hierarchy. In such a scenario, when traffic demands (setup signaling messages) reach the edge of the optical network, the GMPLS-aware *Optical Cross-Connect* (OXC) has to assign them lightpaths and route them by some kind of traffic grooming algorithm. A recent trend in traffic grooming literature has been to consider the emerging area of *dynamic traffic grooming*, in which traffic demands are not considered to form an absolutely static traffic matrix, but may change over time. Different models of variation have been considered, from a completely dynamic call arrival and departure model, to nearly static traffic matrices with a small amount of allowed variations in the magnitude of the traffic components. In [3], we have attempted to describe the different categories of assumptions regarding variation of traffic with time and allowed design choices. Grooming time-varying traffic is becoming an increasingly important research area, because increasing network sophistication is likely to make networks lower in the aggregation hierarchy, where aggregation does not smooth out most variations in traffic, candidates for grooming strategies. Next-generation SONET (NG-SONET) networks are expected to carry time-varying traffic. To overcome the intrinsic inefficiency of carrying dynamic traffic demands, the mechanisms of Virtual Concatenation (VCAT) (as defined in [ITU-T G.707]) and Link Capacity Adjustment Scheme (LCAS) (as defined in [ITU-T G.7042]) have been developed. Using VCAT, non-contiguous Synchronous Payload Envelopes (SPEs) that form a Virtual Concatenation Group (VCG) can be routed on the physical topology independently and recombined at the destination. Complementary to VCAT, LCAS dynamically adjusts the capacity allocated to a traffic component by adding or removing members of its VCG.

A. Prior Work

Recently, studies on dynamic traffic grooming have appeared in the literature. Most of them consider discrete call arrival and departure models, and focus on the blocking probability experienced by sub-wavelength calls as the relevant performance metric.

In [4], the authors study the dynamic traffic grooming problem with rearrangement on ring networks. The authors provide a reconfiguration algorithm, called bridge-and-roll (BR), such that the number of LTEs is reduced while keeping

¹This research was supported in part by NSF grant # ANI-0322107.

the network as bandwidth efficient as a full opaque network. The dynamic traffic grooming problem in Mesh networks using a graph model is studied in [5]. This model creates an auxiliary graph and by assigning different weights to the edges, various objectives using different grooming policies, can be evaluated. The study in [6] also proposes some graph-based dynamic traffic grooming algorithms. The main constraints considered are the number of transceivers and wavelength continuity. In [7]–[9], the performance analysis problem related to dynamic traffic grooming is studied. Queueing models are used to determine the blocking probability under a given call arrival/departure model and network policy. We do not discuss them further here because our focus in this paper is a design problem, rather than one of analysis. In [10], we had proposed the over-provisioning method as an integrated approach to supporting dynamic traffic grooming, by balancing the reconfiguration cost and traffic grooming gain. This also improves network agility, or response time to change in traffic demands. However, the optimality of overprovisioning was not considered in [10]; instead naive rule-of-thumb approaches were used such as all-equal, proportional etc.

In this paper, we focus on a specific spare capacity allocation in support of dynamic grooming. As we show in the next section, our problem can be viewed as a version of the packing problem. In the literature, both the integer version and the continuous versions of this problem have been extensively studied. In [11], Hauchbaum and Shanthikumar present an algorithm for the continuous version, which is polynomial in terms of the input size, required precision and the largest subdeterminant. They also show that for the integer version, the algorithm is polynomial provided that the corresponding integer linear version is polynomial. In [12], instead of using the widely used scale-and-iterate approach, Magnanti and Stratila present a polynomial approximation algorithm using a *single* piecewise linear approximation for each component of the objective function. It is shown that using pieces exponentially increasing in size, the polynomial solvability for separable concave problems is still guaranteed.

B. Our Contribution

In this paper, we study a problem motivated in [10]. Given a current grooming solution in a network in which the traffic components can request increments and decrements at intervals, we address the question of how to pre-allocate any remaining spare capacity to allow the network to continue operating as long as possible before reconfiguring the logical topology in order to satisfy the requests. We have previously considered this question in [13], but only for the case of a single lightpath. In this paper, we show that while this problem can be solved optimally in the single link case [13], the problem becomes difficult for general topologies. We go on to provide good heuristics to perform the allocation in that case.

The rest of this paper is organized as follows. The next Section formulates the problem precisely. After briefly reviewing the relevant results of the single-link case in Section III, in

Section IV we address the general topology case. Section V presents our numerical results, followed by the conclusion of our paper.

II. PROBLEM FORMULATION

We consider an optical network with a logical topology currently implemented, and carrying traffic specified by a traffic demand matrix. The logical topology, the current traffic matrix and the current routing of each component of the traffic matrix on the logical links comprising the logical topology (that is, the complete current grooming solution) are given as part of the problem. Each arc of the logical topology graph is a continuous optical channel, traversing more than one physical fiber link in general, formed by wavelength switching at intermediate nodes. Such end-to-end optical channels have been called “lightpaths” in literature; in this work we refer to them as *logical links*, or simply *links* when the context makes it clear that logical (and not physical) links are under consideration. Each traffic demand component is specified as an integer, indicating multiples of some basic traffic rate (such as OC-3). In the same units, the capacity of each wavelength channel is denoted by C . Each traffic component can undergo some change in values, that is traffic components can request increment and decrement of the current magnitude. An increment request must be satisfied by allocating the increment *with the same routing over the logical topology*, to guarantee the same delay and jitter characteristics to all parts of the traffic flow. In our model, we allow each traffic component to request an increment or decrement of only one unit at a time, but the requests are assumed to come randomly. A decrement request can be satisfied at any time, but an increment request can be satisfied only if the traffic component has currently been overprovisioned by at least one unit. That is, we only count an *agile* admission of an increment request as successful. It is possible that an increment request can be put on hold while the logical topology and grooming is rearranged in order to allow the request to be satisfied, but this involves much higher delays in satisfying the request and we do not count such a request as satisfied, in this paper. Following [10], we assume that reconfigurations can be undertaken only *between* successive requests, and the problem at hand is to decide how to allocate the spare capacity in each link of the logical topology as overprovision to the traffic components traversing that link. Note that the physical topology (and lightpath RWA on the physical topology) can also be considered given, but are not required in our formulation, as a consequence of the above focus. Thus our work is applicable to a broad range of networking situations where traffic flows are routed over a network and spare capacity can be allocated by overprovisioning.

We denote the amount of current traffic demands by the traffic matrix $T = [t_{sd}]$, where t_{sd} is the traffic demand from source node s to destination node d . For any individual traffic component, the increment and decrement of the traffic component are stochastic events, hence the time until an increment request occurs that can not be met is a random variable. We call this variable the *Time To Rejection* (TTR) of that traffic

component. Clearly, the larger the spare capacity assigned to that traffic component, the larger the expected or Mean TTR (MTTR). Now considering the whole network, from the provider's point of view, any rejection of an increment request is undesirable. Thus we define the TTR of the entire traffic matrix to be the time until the first rejection of an increment request occurs for *any* of the traffic components. The goal of spare capacity assignment is to maximize the MTTR for the entire network, *i.e.* the expected time until the first such rejection occurs. Again, it is obvious that the MTTR depends on the spare capacity assignment.

Let $\mathcal{L}_i$ be the set of logical links traversed by traffic component i , and S_l be the spare capacity available on link l . We denote the spare capacity assigned to traffic component i on link l by $s_i^{(l)}$, and the MTTR of the traffic matrix under this spare capacity assignment by $Q(\{s_i^{(l)}\})$. Then, the problem can be formulated as the following optimization problem:

$$\text{maximize} \quad Q(\{s_i^{(l)}\}) \quad (1)$$

$$\text{subject to} \quad \sum_{i: l \in \mathcal{L}_i} s_i^{(l)} \leq S_l, \forall l \quad (2)$$

$$s_i^{(l)} \in Z_0^+, \forall i, l \in \mathcal{L}_i. \quad (3)$$

This problem can be viewed as a special case of the packing problem. It is in general a difficult problem, and we show that this is true of our particular problem in Section IV. However, in Section III, we show that it can be easily solved optimally for a single link, under reasonable assumptions. First, we adopt a traffic variation model and study the effect it has on the MTTR of a traffic component in isolation.

A. Memoryless Variation Model and MTTR

Naturally, the nature of the function $Q(\{s_i^{(l)}\})$ depends on the process governing increment and decrement request arrivals. In general, $Q(\{s_i^{(l)}\})$ is a non-linear function. In the absence of any basis for assuming that increment and decrement requests from the different traffic flows are correlated, we make the natural assumption that the requests from each traffic flow is an independent process. For each traffic component, we assume that the arrival process of increment requests is memoryless with rate γ , and that of decrement requests is similarly memoryless with rate μ . To make the problem general, suppose that there is a lower bound (LB_{sd}) and an upper bound (UB_{sd}) on the amount of traffic component $t_{(sd)}$, that is, t_{sd} must be within the closed interval $[LB_{sd}, UB_{sd}]$, such that $\infty \geq UB_{sd} \geq LB_{sd} \geq 0$. Then a simple Markov model suffices to represent the behavior of any traffic component. Each state t in the closed Markov chain represents the case when the current value of the traffic component is t . Since the chain is defined only on values of $t_{(sd)}$ within the allowed limits, the lower bound serves only to change the labels of the states; and henceforth we only consider $LB_{sd} = 0$ without any loss of generality. Define the matrix $\mathcal{M} = [\mathcal{M}_{i,j}]$, where the $[i,j]$ -th element $\mathcal{M}_{i,j}$ is the *first passage time* from state i to state $j+1$. Note that the first

passage time from a given state is *only defined to higher states*, thus the elements of the matrix below the diagonal do not exist or are not defined. Let s units spare capacity be allocated to this traffic demand (that is, a total of $t+s$ units of capacity is allocated for this demand). It is clear that an increment request can be satisfied as the traffic demand evolves, through increment and decrement requests, as long as the state is below $t+s$. However, an increment request when the current state is $t+s$ cannot be satisfied and will cause the first rejection. Note that the time to rejection becomes infinity if the amount of spare capacity is assigned such that $t+s \geq UB_{sd}$. For notational convenience, we denote $\mathcal{M}_{i,j}$ by $m(i, j-i)$. This allows us to conveniently represent the mean time until a traffic component in state t and allocated s units of overprovision grows beyond its provisioning as $m(t, s)$. It is now obvious that the quantity $m(t, s)$ represents the MTTR of the traffic component.

From the Markov model, it is obvious that $m(0, 0) = \frac{1}{\gamma}$. We can then find the further elements of $\mathcal{M}$ by the following recursive arguments. First, we observe that:

$$m(t, s) = m(t, s-1) + m(t+s, 0) \quad \forall t \geq 0, s > 0, \quad (4)$$

which simply states that the mean first passage time from state t to $t+s+1$ is equal to the mean first passage time from state t to $t+s$ plus the mean first passage time from state $t+s$ to $t+s+1$. The mean first passage time from state t to $t+1$ can be derived as follows:

$$m(t, 0) = \frac{1}{\mu + \gamma} + \frac{\mu}{\mu + \gamma} m(t-1, 1), \quad (5)$$

where the first term is the mean sojourn time at state t , the second term is the probability that there is a departure times the mean first passage time from state $t-1$ to $t+1$. Using (4) we obtain

$$m(t-1, 1) = m(t-1, 0) + m(t, 0). \quad (6)$$

Finally, substituting (6) into (5),

$$m(t, 0) = \frac{1}{\gamma} + \frac{\mu}{\gamma} m(t-1, 0) \quad \forall t > 0. \quad (7)$$

Thus we obtain :

$$m(t, s) = \frac{s+1}{\gamma} + \frac{(s+1)\mu}{\gamma^2} + \dots + \frac{(s+1)\mu^t}{\gamma^{t+1}} + \frac{(s+2)\mu^{t+1}}{\gamma^{t+2}} + \dots + \frac{\mu^{t+s}}{\gamma^{t+s+1}} \quad (8)$$

$$= \frac{s+1}{\gamma} \sum_{i=1}^t \left(\frac{\mu}{\gamma}\right)^i + \frac{\left(\frac{\mu}{\gamma}\right)^t}{\mu} \left((s+2) \sum_{i=1}^{s+1} \left(\frac{\mu}{\gamma}\right)^i - \sum_{i=1}^{s+1} \left(\frac{\mu}{\gamma}\right)^i i \right) \quad (9)$$

While the above allows us to find the mean of the TTR, we also need its distribution to proceed. The distribution of first passage time is a complex and well-investigated topic. For our purposes, we are interested in the practical result that under many reasonable circumstances, the first passage

time is well approximated by an exponential distribution. This result was first shown in [14], and has been used in many different contexts since. The method requires that the mean of the distribution be found independently of the exponential approximation assumption, as we have done above. We have also verified by numerical experimentation that the approximation holds well for our problem, some such numerical results were presented in [13]. We proceed with this approximation.

III. THE SINGLE LOGICAL LINK CASE

We considered the case for a single logical link in [13], and presented several algorithms, which we proved were all equivalent and optimal. The practical utility of such an algorithm is limited in itself, because it only applies to a single lightpath considered in isolation. However, we can use such an algorithm as a building block for a general network with many lightpaths, which is our contribution in this paper. For the sake of context, we briefly describe one of the algorithms from [13] in this section.

A. A Convex Objective Function

To show that the single-link problem is a convex optimization problem, we consider a continuous function analogous to the MTTR. We denote the continuous analogue of the discrete variable s as σ , of t as τ , and let $m(\tau, \sigma)$ denote the MTTR for a traffic component of magnitude τ to which σ spare capacity is allocated. We also denote $r = \mu/\gamma$. Since only one link is being considered, we simplify the notation for the spare capacity assignment to s_i instead of $s_i^{(l)}$, similarly we speak only of σ_i .

If $r \neq 1$, by substitution in (8) and using standard geometric summation identities, we obtain

$$m(\tau, \sigma) = \frac{1}{\gamma(r-1)^2} [(\sigma+1)(1-r) + r^{\tau+1}(r^{\sigma+1} - 1)]. \quad (10)$$

If $r = 1$, it is easy to show, again by substitution in (8),

$$m(\tau, \sigma) = \frac{(\sigma+1)(\sigma+2\tau+2)}{2\gamma}. \quad (11)$$

We consider a link l , and denote the set of traffic components traversing this link by $\mathcal{T}$ and $n = |\mathcal{T}|$. Let τ_i be the current amount of traffic demand $i, i \in \{1 \dots n\}$. Let the total amount of spare capacity on the link be S_l , that is $S_l = C - \sum \tau_i$. The problem is to determine the spare capacity σ_i to be assigned to each traffic component so that the MTTR for the whole link is maximized. Because spare capacity is assigned individually to each traffic component and not shared, the TTRs for the individual traffic components are independent random variables. From the last section, we know that each TTR can be considered to be exponentially distributed. By a well-known result, the minimum of independent exponentially distributed random variables is also an exponentially distributed random variable, with rate equal to the sum of the individual rates [15]. Thus the MTTR of the

set of traffic components traversing the link l is given by the harmonic sum of the individual MTTRs:

$$Q_l(\vec{\sigma}) = \frac{1}{\sum_{i=1}^n \frac{1}{m_i(\tau_i, \sigma_i)}} = \frac{1}{z_l(\vec{\sigma})}. \quad (12)$$

Thus the optimization problem, in terms of the continuous quantities, is simply to *minimize* the *rejection rate* $z_l(\cdot)$. This is a non-linear objective function, but fortunately, it is an easy one in the practical sense, as the proposition below shows.

Proposition 1: The objective function z_l as given by (12) is a strictly convex function of $\vec{\sigma} = [\sigma_1, \dots, \sigma_n]$.

The proof of the above is tedious but straightforward, and can be found in full in [13]. Briefly, it consists of showing that $1/m_i(\tau_i, \sigma_i)$ is convex, which is obvious for $r = 1$ and can be shown for $r \neq 1$ by taking the second partial derivative and decomposing it into parts until every part can be shown to be positive or zero. Then $z_l(\vec{\sigma})$ is convex separable in the sense that it is the sum of convex functions $1/m_i(\tau_i, \sigma_i)$, each σ_i is an independent coordinate. The convexity depends on the assumption we made regarding the state independence of γ and μ . We have investigated this issue in detail under other conditions, and can report the following: convexity is retained even when the increment request rate and decrement request rates γ_t and μ_t are different for different states t , if $r_t = \mu_t/\gamma_t < 1$ for all states t , and $m_{t-1, t-1} \geq \frac{1}{\gamma_t - \mu_t}$. If either condition is violated in the general case of state-dependent γ_t and μ_t , then instances exist that show the objective is neither convex nor concave. Fortunately, our assumption of state-independence is quite reasonable, and indeed perhaps the most natural assumption in the absence of any specific knowledge about the nature of traffic variation with time, so our conclusions continue to be valuable. We proceed with this assumption.

B. An Optimal Algorithm

Below we present a greedy algorithm for the allocation of the total slack S_l to the different traffic components traversing the lightpath. We use d_i to denote the i -th element of a vector $\vec{d}$ of n elements. Let $\vec{d}'$ denote a vector such that $\sum_i d'_i = 0$, and $\vec{d}_k$ denote a vector of n elements for which the k -th element is 1 and all other elements are zero. Below we use these as follows: $\vec{d}'$ denotes a rearrangement of spare capacity among the n components, and $\vec{d}_k$ denotes the addition of one new unit of spare capacity to some traffic component.

Algorithm Alloc-1-link:

- 1) Set $\tilde{s} : s_i = 0 \quad \forall i$
- 2) For $x = 1$ to S_l ,
- 3) $\tilde{s} \leftarrow \tilde{s} + \vec{d}_k : k = \operatorname{argmin}_l z_l(\tilde{s} + \vec{d}_l), \forall l$

Obviously, this algorithm will terminate. The following proposition asserts that $\tilde{s}$ gives us an optimal solution at the termination of the algorithm.

Proposition 2: Let $\tilde{s}$ be the solution given by algorithm **Alloc-1-link**. Then, we have

$$z_l(\tilde{s}) \leq z_l(\tilde{s} + \vec{d}') \quad \forall \vec{d}'. \quad (13)$$

We omit the proof for the sake of brevity. It can be found in [13], and is a proof by induction; we show that after the c -th execution of Step 3, $\tilde{s}$ comprises an optimal allocation of c units of spare capacity to the n traffic components. At the first step, it is easy to see that algorithm **Alloc-1-link** will provide us the optimal solution, therefore (13) holds and the basis is obtained. At a step $c \in [2, S_l]$, the convexity we have proved earlier allows us to show that this optimality is maintained. We mention that when $UB_{sd} \neq \infty$ for some $t_{(sd)}$, the first passage time $m(t, s)$ becomes infinite for any spare capacity allocation $s \geq UB_{sd} - t$. This means that the last piece of the piecewise linear representation of $1/m(t, s)$ abruptly goes to zero, possibly causing convexity to be lost in this region. However, this is easy to correct for algorithmically, and we omit details of this straightforward procedure.

IV. THE GENERAL TOPOLOGY CASE

In a realistic traffic grooming network, there will be a logical topology consisting of a large number of links, and end-to-end traffic components will traverse more than one link in general. In this section, we consider this more practical multiple-link problem.

A. Complexity of the General Case

In multi-hop networks, the same traffic component may traverse more than one logical links. When the traffic component requests an increment, the request can be satisfied only if it has a unit of overprovisioned spare capacity remaining on every link it traverses. Therefore its MTTR is determined by its spare capacity allocation on the link on which it has the smallest spare capacity allocation. Thus larger spare capacity allocated to this component on other links is wasted, but this becomes apparent only when the different links are considered together. Thus coupling is introduced in the problem, and we suspect that the problem becomes conceptually much harder in this case. In this subsection, we show that this is indeed true. First, we present a slightly simpler formulation of the general case.

In the function $m(t, s)$ (more simply, $m(s)$), the only quantity that depends on the link under consideration is the spare capacity. Thus, using the terminology we defined in Section II, it is easy to see that the MTTR M_i of a traffic component i traversing the set of lightpaths $\mathcal{L}_i$ is given by:

$$M_i = \min_{l \in \mathcal{L}_i} (m_i(s_i^{(l)})) = m_i \left(\min_{l \in \mathcal{L}_i} s_i^{(l)} \right). \quad (14)$$

The overall objective now is to maximize the time until the first rejection of an increment request in the whole network. The MTTR of the whole network is again seen to be the harmonic sum of the MTTRs of the all the traffic components, as a consequence of the result regarding the minimum of exponentially distributed variables referred to in Section II.

Hence the objective function of the general mathematical programming formulation in (1) can be obtained as:

$$Q(\{s_i^{(l)}\}) = \frac{1}{\sum_{i=1}^n \frac{1}{m_i(\min_{l \in \mathcal{L}_i} s_i^{(l)})}} = \frac{1}{z(\{s_i^{(l)}\})}, \quad (15)$$

and again, the problem can be posed as one of minimization of the rejection rate $z(\{s_i^{(l)}\})$, now for the whole network.

As we stated above, in any solution, a traffic component may be allocated more spare capacity on some links than others; this allocation is not useful and could be removed without changing the rejection rate achieved by the solution. This is true of optimal solutions also. If we relax the requirements that all the spare capacity on every link must be assigned to some traffic component or other, we can abandon separate variables $s_i^{(l)}$ for the slack allocation received over different links by the same traffic component i , and use a unique variable s_i to represent the spare capacity allocation received by this traffic component over each link it traverses. This allows us to obtain the following simpler formulation:

$$\text{minimize} \quad \sum_i \frac{1}{m_i(s_i)} \quad (16)$$

$$\text{subject to} \quad \sum_{i: l \in \mathcal{L}_i} s_i \leq S_l \quad (17)$$

$$s_i \in \mathcal{Z}_0^+, \forall i. \quad (18)$$

But note that we had to use an inequality in (17).

Proposition 3: The problem of minimizing rejection rate by spare capacity assignment is NP-Hard in a general topology network.

Proof: Our proof follows a derivation in [16], which is a study on the maximum integral multicommodity flow problem. We reduce the NP-Hard three-dimensional matching problem to the spare capacity allocation problem. An instance of the three-dimensional matching problem [17] is specified by three disjoint sets $X = \{x_i\}, Y = \{y_i\}, Z = \{z_i\}, i = 1 \dots n$, and a set of triples $S = \{x_i, y_i, z_i\}$; the problem is to find the maximum number of disjoint triples. We construct an instance of the spare capacity assignment problem as follows. The network is a tree T with with height 3, the root is marked as r . At level 1, there are $3n$ nodes representing the elements in $X \cup Y \cup Z$, labelled as $x_i, y_i, z_i, i = 1 \dots n$ correspondingly. Between each node x_i and the root, there is a bidirectional link, with 1 unit of spare capacity on each direction. From the root to each node z_i , there is a directed link with 1 unit spare capacity. From each node y_i to the root, there is a directed link with 1 unit spare capacity. Each node $x_i, i = 1 \dots n$ has p_i children, where p_i is the number of occurrence of x_i in S . These nodes are labelled as $x_{i,l}, 1 \leq l \leq p_i$. There is a bidirectional link between x_i and each $x_{i,l}$, with 1 unit spare capacity on each direction. Each $x_{i,l}$ has two children, left child labelled as $x_{i,l,a}$, right child $x_{i,l,b}$. There is a directed link from $x_{i,l,a}$ to $x_{i,l}$ and a directed link from $x_{i,l}$ to $x_{i,l,b}$, each with 1 unit spare capacity.

Next we define the traffic components. For easy readability, we denote a traffic component from s to d with the notation $(s \rightarrow d)$ or $(d \leftarrow s)$ in what follows. We introduce traffic $(x_{i,l,a} \rightarrow x_{i,l,b}), \forall i, l$ and $(x_{i,l,a} \rightarrow z_k), (x_{i,l,b} \leftarrow y_j)$ if $(x_i, y_j, z_k) \in S$. All traffic components have the same existing amount of traffic demand t and arrival rate γ and departure rate μ ; thus all have the same MTTR function $m(\cdot)$. An example

of this construction is shown in Fig. 1, where solid black lines represent logical links and shaded lines represent traffic demand components.

Fig. 1. Construction of tree network from 3DM

The amount by which the objective function $z(\{s_i^{(l)}\})$ is reduced for one unit of spare capacity assigned to any traffic component is $\Delta = \frac{1}{m(1)} - \frac{1}{m(0)}$. Because the spare capacity available for each traffic component is at most 1 unit, the objective function value is minimized if and only if the maximum amount of spare capacity can be assigned. Now we show that the instance of three-dimensional problem has t disjoint triples if and only if $t + |S|$ units of spare capacity can be assigned in the corresponding spare capacity allocation problem instance constructed. Consider a solution $S' \subseteq S$ to the 3DM problem instance with t disjoint triples. If x_i is not covered by S' , we assign 1 unit spare capacity to traffic components from $x_{i,l,a} \rightarrow x_{i,l,b}$, $\forall 1 \leq l \leq p_i$. Otherwise, if $(x_i, y_j, z_k) \in S'$ and corresponds to the l th occurrence of x_i , then we assign 1 unit spare capacity to the traffic components $(x_{i,l,a} \rightarrow z_k)$ and $(x_{i,l,b} \leftarrow y_j)$ respectively, $\forall m \neq l$. Thus, a covered x_i results in an assignment of $p_i + 1$ unit of spare capacity, and an uncovered x_i results in an assignment of p_i unit of spare capacity. Therefore, the total amount of spare capacity assigned is $t + \sum_{i=1} p_i = t + |S|$.

Conversely, suppose that $t + |S|$ units of spare capacity can be assigned. Note that the maximum amount of spare capacity can be assigned over traffic components that have at least one end in the subtree rooted at x_i is $p_i + 1$. Moreover, the assignment can only be done by giving 1 unit spare capacity to traffic components $(x_{i,l,a} \rightarrow z_k)$ and $(x_{i,l,b} \leftarrow y_j)$, for some l , and 1 unit to each of the remaining $p_i - 1$ traffic components, $(x_{i,m,a} \rightarrow x_{i,m,b})$, $\forall m \neq l$. Since $t + |S|$ units of spare capacity are assigned, there must be at least t elements x_i such that $p_i + 1$ units spare capacity are assigned to traffic components that have at least one end in the subtree rooted at x_i . If spare capacity is assigned to traffic components $(x_{i,l,a} \rightarrow z_k)$ and $(x_{i,l,b} \leftarrow y_j)$, then (x_i, y_j, z_k) is a triple in S . Let S' be the set of these (at least t) triples, each x_i, y_j, z_k is included only once because of the amount of spare capacity on the edges between the root and nodes at level 1.

The transformation was carried out in polynomial time, and we have shown that solutions to each problem can be transformed into solutions for the other in polynomial time as well. Since the 3DM problem is NP-Hard, so is the spare capacity allocation problem. ■

In view of the NP-hardness of the problem, in the following subsections, we introduce heuristic algorithms.

B. A Heuristic Based on Relaxation

We develop a heuristic algorithm based on Lagrangian relaxation, as in [18]. We use the following notations: $\lambda = \{\lambda_l, l = 1, \dots, m\}$, where m is the number of links, is the vector of lagrangian multipliers. $\tilde{S} = \{s_i, i = 1, \dots, n\}$ is the spare capacity allocation vector, where s_i is the number of units of spare capacity assigned to traffic component i .

The algorithm is conceptually in two parts: introducing *fictitious* traffic to remove the inequality constraint of (17), and then using the standard technique of Lagrangian relaxation. The heuristic (as opposed to exact) nature of the algorithm lies in the first part; as we show, the duality gap in the Lagrangian relaxation is zero in this case. We also show that relaxing the integrality constraint does not affect the integrality of the solution, for this reason we continue to use the integer spare capacity variables s rather than the continuous variables σ in this section. Thus, the solution obtained from the Lagrangian relaxation will always be an exactly optimal solution to the problem with the fictitious traffic, but may not always be optimal for the original problem.

In (16)-(18), we simplified the formulation at the cost of introducing an inequality in the place of an equality for the total spare capacity allocated. Effectively, the introduction of fictitious traffic is to regain the advantage of equality. For every link with spare capacity, we add a fictitious traffic component traversing only that single link *if* there is no such traffic component in the given problem instance itself. The current magnitude of the fictitious traffic components are zero and the increment request rates γ are set to a very small value. This ensures that the integral optimal solution to this modified problem must be a solution such that all units of spare capacity are assigned; obviously, in any spare capacity assignment that leaves unassigned units of spare capacity on some link l , we can decrease the objective function value by assigning the unassigned spare capacity to the traffic component that only traverses l . The primal problem is then as follows:

$$\text{minimize} \quad Z(\tilde{S}) = \sum \frac{1}{m_i(s_i)} \quad (19)$$

$$\text{subject to} \quad \sum_{i:l \in \mathcal{L}_i} s_i = S_l, \forall l \quad (20)$$

$$s_i \geq 0, \forall i \quad (21)$$

The lagrangian dual problem is to *maximize* the quantity:

$$\min_{\tilde{\lambda}} L(\tilde{\lambda}) = - \sum_i \frac{1}{m_i(s_i)} + \sum_l \lambda_l \left(\sum_{i:l \in \mathcal{L}_i} s_i - S_l \right) \quad (22)$$

We can neglect the non-negativity constraints for variables $s_i, i = 1, \dots, n$, by incorporating them into the cost function,

i.e., setting the value of the cost function as infinity when an s_i reaches 0^- . In solving the dual, we can also ignore the integrality constraints on spare capacity variables; as we see below in Figure 2 and the discussion following, an integer solution will always be obtained if the piecewise linear representations (conceptually defined on the continuous domain) are used. It is easily shown that the duality gap for this Lagrangian relaxation is zero. Briefly, suppose we have a primal feasible solution $\tilde{S}^*$ that is optimal to the primal problem (19)-(21), and a dual solution $\tilde{\lambda}^*$ that is optimal to the lagrangian dual problem (22). Since the second term of equation (22) is always zero, $Z(\tilde{S}^*) = L(\tilde{\lambda}^*)$.

It is shown in [18] that a primal feasible solution $\tilde{S}^*$ satisfying constraint (20) is optimal if and only if there exists $\tilde{\lambda}$ such that $\sum_{l:l \in \mathcal{L}_i} \lambda_l$ is sub-gradient at s_i^* , for each traffic component i . Equivalently, it is optimal if and only if:

$$s_i^* = \operatorname{argsup}_{s_i} \left(\sum_{l:l \in \mathcal{L}_i} \lambda_l s_i - \frac{1}{m_i(s_i)} \right) \quad \forall i \quad (23)$$

is satisfied. Fig. 2 illustrates the nature of this function. From the figure, because $-\frac{1}{m_i(s_i)}$ is an increasing function,

Fig. 2. An illustration of the function

$\sum_{l:l \in \mathcal{L}_i} \lambda_l s_i - \frac{1}{m_i(s_i)}$ is unbounded if $\sum_{l:l \in \mathcal{L}_i} \lambda_l \geq 0$, and it attains the unique supremum at $s_i^* : \sum_{l:l \in \mathcal{L}_i} \lambda_l \in \vartheta \left(\frac{1}{m_i(s_i^*)} \right)$, if $\sum_{l:l \in \mathcal{L}_i} \lambda_l < 0$, where $\vartheta \left(\frac{1}{m_i(s_i^*)} \right)$ is the set of sub-gradient of function $\frac{1}{m_i(s_i)}$ at s_i^* . Since $\frac{1}{m_i(s_i)}$ is monotonically decreasing, given the vector $\tilde{\lambda}$, $\tilde{S}$ can be easily obtained by searching techniques. Moreover, Fig. 2 shows that s_i is an increasing function of $\sum_{l:l \in \mathcal{L}_i} \lambda_l$ because as $\sum_{l:l \in \mathcal{L}_i} \lambda_l$ gets smaller, the supremum moves towards 0. This also allows us to observe that the best integer solution (better than neighboring integer solutions) must necessarily be better than non-integer solutions in the near vicinity, because of the piecewise linear nature of the function. Thus integer solutions will be obtained even if integrality constraints are not explicitly enforced, as we remarked above.

We define infeasibility in the dual as follows. Since $\frac{\partial L(\tilde{\lambda})}{\partial \lambda_l} = \sum_{i:l \in \mathcal{L}_i} s_i - S_l, \forall l$, if the solution is primal feasible, we have

$\frac{\partial L(\tilde{\lambda})}{\partial \lambda_l} = 0, \forall l$. We call $|\beta| = \left| \frac{\partial L(\tilde{\lambda})}{\partial \lambda_l} \right|$ the infeasibility of the constraint on link l , if $\beta \neq 0$.

We are now ready to describe the relaxation algorithm. The superscript r is used to represent the variables at the start of the r -th iteration and $\delta \in (0, 1)$ is a predefined real number. In our implementation, we choose a coordinate (traffic component) c in a cyclical manner in the iterations, and Δ_c is found by binary search.

Algorithm Alloc-Relax:

- 1) repeat
- 2) find $\tilde{S}^r : s_i^r = \arg \sup_{s_i} \left(\sum_{l:l \in \mathcal{L}_i} \lambda_l^{r-1} s_i - \frac{1}{m_i(s_i)} \right) \quad \forall i$
- 3) compute $\frac{\partial L(\tilde{\lambda})}{\partial \lambda_l} : \frac{\partial L(\tilde{\lambda})}{\partial \lambda_l} = \sum_{i:l \in \mathcal{L}_i} s_i^r - S_l, \forall l$
- 4) choose any coordinate c , set $\beta = \frac{\partial L(\tilde{\lambda})}{\partial \lambda_c}$
 - a) if $\beta = 0$, do nothing
 - b) if $\beta > 0$, $\lambda_c^r = \lambda_c^{r-1} - \Delta_c : 0 \leq \frac{\partial L(\tilde{\lambda})}{\partial \lambda_c} \leq \delta \beta$
 - c) if $\beta < 0$, $\lambda_c^r = \lambda_c^{r-1} + \Delta_c : 0 \geq \frac{\partial L(\tilde{\lambda})}{\partial \lambda_c} \geq \delta \beta$
- 5) until gain in objective stabilizes, OR maximum number of iterations is exceeded

In each iteration of the algorithm, the optimality condition (23) is maintained, and the infeasibility β is reduced by updating $\tilde{\lambda}$. It is always possible to so reduce the infeasibility because of the following reason. Suppose for a chosen coordinate c , $\beta > 0$ (respectively, $\beta < 0$), λ_c^{r-1} is decreased by Δ_c (respectively, increased by Δ_c); since $s_i^{r+1} \forall i : c \in \mathcal{L}_i$ is increasing function of λ_c^r , the infeasibility $|\beta|$ is also reduced.

Next, we briefly analyze the effect of fictitious traffic components. The basic idea is that by adding fictitious traffic components, we can turn the optimization problem with inequalities to an optimization problem with equalities, which is conceptually easier to solve. We found that after relaxing the equality constraints, the integer and non-negativity constraints of the primal problem are redundant because the optimal solution of the Lagrangian dual problem is always integral and non-negative for a given dual vector. Note that this is different from adding slack variables to the original problem because the slack variables have non-negativity constraints, which cannot be omitted in the dual problem. Moreover, the dual variables are free (with no requirements on their signs), thus, the Lagrangian dual problem is an unconstrained problem, which is easy to solve.

However, by adding fictitious traffic components, the original problem has been modified. The difference is interesting in terms of the well-known Karush-Kuhn-Tucker (KKT) conditions. In the original problem, we have the complementary slackness condition, that is, if on a link, the optimal solution does not claim all the spare capacity (the sum of spare capacity assigned to traffic components traversing the link is strictly less than the total amount of spare capacity, suppose the amount of unused capacity is s_u), the corresponding optimal dual variable of the link must be zero. However, after adding fictitious traffic components, because of the equality constraints, the complementary slackness condition is always satisfied. Now, suppose that the optimal solution to the original problem is also optimal to the modified problem. It turns out that the

dual variable on link l must also be a sub-gradient of the cost function of the fictitious traffic component on link l at point s_u . Since that the arrival rate of the fictitious traffic component is very low, the sub-gradient at s_u is an interval very close to zero. Therefore, instead of being zero in the original problem, the dual variable is relaxed to be in an interval very close to zero if fictitious traffic components are added.

C. Algorithm **Alloc-Greedy**: greedy heuristics

The above algorithm has good performance but is somewhat complex. In some cases, it may be suitable for implementation for on-line application in the network. In other cases, a simpler approach may be desired. For such cases, we propose three different flavors of a very simple greedy approach in this section.

Basic Greedy Approach (GA:) First, we propose the basic greedy approach. We use the optimal algorithm developed for the single-link model on each link. Because a traffic component can be assigned different amounts of spare capacity on different links it traverses, we pick the minimum amount and assign this amount of spare capacity to the traffic component on every link it traverses. After doing this for all traffic components, we may have some unused capacity left on each link. We simply assign it to single-link traffic if it exists. The algorithm is as follows:

- 1) Let S_l be the available spare capacity on link l , use Algorithm **Alloc-1-link** for each link
- 2) Let $\mathcal{P} = \{i : |\mathcal{L}_i| > 1\}$, for each i do
- 3) $s_i = s_i^{(k)} : k = \arg\min_{l \in \mathcal{L}_i} \frac{1}{m_i^{(l)}(s_i^{(l)})}$,
- 4) $S_l \leftarrow S_l - s_i^{(k)}, \forall l \in \mathcal{L}_i$
- 5) For each l , allocate S_l to all traffic components traversing l and $\notin \mathcal{P}$ using Algorithm **Alloc-1-link**.

This algorithm is heuristic in the sense that a traffic component can lend its over-allocated spare capacity to other single-hop traffic components, since this will benefit others without sacrificing itself. However, it is possible that borrowing some spare capacity from other traffic components on the critical link, instead of lending over-allocated spare capacity to other traffic components on other links, is more beneficial. That is, the amount of spare capacity allocated on the critical link is not necessarily a lower bound of the optimal allocation of the traffic component. In what follows, we will discuss a modified version that involves a more complicated lending procedure.

Iterative Algorithm (IA:) This algorithm is a modification of the previous one.

- 1) Initialize $s_i = 0 \forall i$
- 2) while $\exists l : S_l > 0, \forall l$
- 3) let S_l be the available spare capacity on link l , use Algorithm **Alloc-1-link** for each link
- 4) let $\mathcal{P} = \{i : |\mathcal{L}_i| > 1\}$, for each i do
- 5) $s_i = s_i + s_i^{(k)} : k = \arg\min_{l \in \mathcal{L}_i} \frac{1}{m_i^{(l)}(s_i^{(l)})}$,
- 6) $S_l \leftarrow S_l - s_i^{(k)}, \forall l \in \mathcal{L}_i$
- 7) end while

The basic idea is that, using the algorithm for the single link case, we may assign different amount of spare capacity on

different links for a traffic component traversing more than one link. However, instead of simply assigning the traffic component the minimum amount of assignment over all links and giving the spared capacity (if exists) to single link traffic components, we use an iterative approach. At each iteration, this modified version reserves the minimum assignment for the traffic component, that is, removes that amount from the spare capacity, then repeats the single link algorithm. The iteration stops when no improvement can be made. Obviously, this algorithm is more complicated than the previous algorithm because, after an amount of spare capacity is reserved for a traffic component, it can still take part in the competition of gaining more spare capacity.

Weighted Iterative Algorithm (WIA:) This is a length-weighted variation to the previous iterative algorithm. Consider the case that, when we use the greedy approach on a single link, we greedily pick a traffic component that provides the largest reduction of the objective function value if 1 unit spare capacity is assigned. In the multi-link case, when two traffic components traversing a link provide the same amount of reduction, we break the tie by choosing the shorter one so that the saved spare capacity on some link can be assigned to other traffic components. In this heuristic, we formalize that intuition by defining a *weighted reduction* of the objective function for each traffic component, as the number of the links it traverses divided by the actual reduction of the objective function value if 1 unit spare capacity is assigned. This provides us a length-weighted approach.

V. NUMERICAL RESULTS

For the single logical link case, we compare the results of algorithm **Alloc-1-link** with a brute-force algorithm that finds the optimal by enumerating all the possibilities. The numerical result verifies that the algorithm is optimal as we have concluded. This is not unexpected, and we do not produce any of those results here.

For the general topology, we generate random logical topologies as follows.

- 1) Input number of nodes n , number of arcs m and k , which satisfy $n + 2k \leq m$.
- 2) Put vertices randomly in k sets, $V_1, \dots, V_k$ with cardinalities $n_1, \dots, n_k$. Randomly select unassigned nodes and assign to the different sets in rotation.
- 3) Within each subset V_j , randomly permute the members, then add arcs to create a directed circuit in the random order obtained.
- 4) Within each set V_j , randomly select two nodes a_i and b_i . Add arcs $(a_1, b_2), (a_2, b_3), \dots, (a_{k-1}, b_k)$.
- 5) Randomly select pairs of nodes and add the arc between them if not already in the network until the total number of arcs is m .

The capacity of each link C is set to 16. We also generate traffic matrices with sub-wavelength traffic components. The shortest path routing algorithm is used to route the traffic matrix on the logical topology. For the sake of simplicity, the traffic matrices are generated such that using the shortest

path routing will not violate the capacity constraint of the link. Specifically, we generate sub-wavelength traffic demands randomly in the interval $[1, 16]$. If a link capacity is violated, all the traffic components traversing the link are regenerated from an interval $[1, 15]$, etc. We apply all algorithms on each randomly generated instance. For each instance, each algorithm generates an assignment of the spare capacity on each link. The MTTR of the network is computed according to the assignment.

To obtain a baseline solution for each instance generated, we solve the continuous version to find the *optimal continuous solution*. This serves as a lower bound on the optimal integer solution. Ideally we should compare the performance of our algorithms with the optimal integer solution, but it turned out to be computationally impossible to obtain actual numerical results for the optimal. Because of the convexity property, the overall problem can be solved in reasonable time by standard nonlinear programming methods, as long as the integer constraints are relaxed.

We generated hundreds of instances and plot the results in the following figures according to different view points. Only the results normalized to the lower bound are plotted. All the figures use MATLABTM boxplots, which allow the representation of a large amount of data concisely. Briefly, each box represents a population of data, the lines indicate the lower quartile, median, and upper quartile values. The long protruding lines show the extent of data, with '+' indicating outliers. More details can be found in [19].

First, we study networks with different sizes. Fig.3 and 4 show the results on randomly generated networks with 5 – 10 and 11 – 16 nodes respectively. Secondly, we study networks with different maximum r , which reflects how fast traffic changes, in all traffic components. Fig.5 and 6 show the results on networks with maximum $r \geq 20$ and < 20 respectively. Thirdly, we study networks with different minimum r in all traffic components. Fig.7 and 8 show the results on networks with minimum $r \geq .2$ and $< .2$ respectively.

Fig. 3. Numerical results: small networks ($N \in [5, 10]$)

Our computational experience shows that in general, the relaxation algorithm runs slower than other heuristics, but

Fig. 4. Numerical results: larger networks ($N \in [11, 16]$)

Fig. 5. Numerical results: maximum $r \geq 20$

faster than finding the continuous optimal. The main reason is that, using the piecewise linear objective function, for a given dual vector, the optimal $s_i, i = 1, \dots, n$ can be easily found in logarithmic time in terms of the number of pieces. Moreover, the solution generated by the relaxation algorithm is generally better than that of other approaches. However, in our implementation, we have a maximum number of iterations. If the program stops before it reaches the maximum number of iterations, it generates a feasible solution because it satisfies all the primal constraints. In addition, the Lagrangian dual problem is always a lower bound of the primal problem, thus, this solution is also optimal to the primal. However, we find that in some cases, the algorithm does not quit until the maximum number of iterations is reached. In these cases, the solutions may be infeasible to the primal. This is the cause for the lower extreme of the boxplot extending below the lower bound in many of the results. Such solutions would need some adjustments to make them feasible, which shows an advantage of the straightforward greedy approaches. The greedy approaches in general perform very well, which is practically a very welcome result. In general, the iterative approaches work better than the simple approach. Since the greedy algorithms

Fig. 6. Numerical results: maximum $r < 20$

Fig. 8. Numerical results: minimum $r < .2$

Fig. 7. Numerical results: minimum $r \geq .2$

are computationally cheap, a practical strategy might be to run them all and adopt the one providing best results.

VI. CONCLUSION

We have investigated a problem of spare capacity allocation in support of dynamic traffic grooming in optical networks. We have formulated the problem as a mathematical programming problem, and shown that it is a separable convex problem for the single logical link case, and a simple algorithm to obtain the optimal integer solution exists. We have shown that this tractability is lost for a general logical topology, and have provided good practical heuristics for that case. Numerical results show that our approaches produce good results.

There are many directions for future work in this area, such as finding other algorithms for the general topology case, addressing the simplifying assumptions we have made such as state independence, etc.; and we are following some of them. We hope to report more results in this area soon.

REFERENCES

[1] R. Dutta and G. Rouskas, "A survey of virtual topology design algorithms for wavelength routed optical networks," *Optical Networks Magazine*, vol. 1, no. 1, pp. 73–89, Jan 2000.

[2] —, "Traffic grooming in WDM networks: Past and future," *IEEE Network*, vol. 16, no. 6, pp. 46–56, Nov/Dec 2002.

[3] S. Huang and R. Dutta, "Research problems in dynamic traffic grooming in optical networks," in *Proceedings of First International Workshop on Traffic Grooming, San Jose, October, 2004, co-located with BroadNets 2004*, 2004, (available from the author website).

[4] R. Kandula and G. Sasaki, "Grooming of dynamic tributary traffic in wdm rings with rearrangements," in *39th Annual Allerton Conference on Communication, Control, and Computing*, Monticello IL, October 2001.

[5] H. Zhu, H. Zang, K. Zhu, and B. Mukherjee, "Dynamic traffic grooming in WDM mesh networks using a novel graph model," *Networking, IEEE/ACM Transactions on*, vol. 11, 2003.

[6] H. Wen, R. He, L. Li, and S. Wang, "Dynamic traffic-grooming algorithms in wavelength-division-multiplexing mesh networks," *Journal of Optical networking*, vol. 2, no. 4, 2003.

[7] C. Xin, C. Qiao, and S. Dixit, "Traffic grooming in mesh WDM optical networks – performance analysis," *Journal on Selected Area of Communications, IEEE*, vol. 22, no. 9, pp. 1658–69, nov 2004.

[8] R. Srinivasan and A. Somani, "Analysis of multi-rate traffic in WDM grooming networks," in *Proceedings of Computer Communications and Networks Conference*, Oct 2002, pp. 296–301.

[9] A. Washington and H. Perros, "Call blocking probabilities in a traffic groomed tandem optical network," *Special issue dedicated to the memory of Professor Olga Casals, Blondia and Stavrakakis (Eds.) Journal of Computer Networks*, vol. 45, 2004.

[10] R. Mahalati and R. Dutta, "Reconfiguration of grooming optical networks," in *BroadNet'04*, 2004.

[11] D. S. Hochbaum and J. G. Shanthikumar, "Convex separable optimization is not much harder than linear optimization," *Journal of the ACM (JACM)*, vol. 37, no. 4, 1990.

[12] T. L. Magnanti and D. Stratila, "Separable concave optimization approximately equals piecewise linear optimization," *Integer Programming and Combinatorial Optimization X*, 2004.

[13] S. Huang and R. Dutta, "On optimal single link allocation of spare capacity for dynamic traffic grooming," in *Proceedings of the International Conference on Telecommunication Systems - Modeling and Analysis*, Dallas, TX, Nov. 2005, pp. 255–264.

[14] J. Keilson, *Markov Chain Models - Rarity and Exponentiality*. Springer-Verlag, 1979.

[15] R. Nelson, *Probability, Stochastic Processes, and Queueing Theory*. Springer-Verlag, 1995.

[16] N. Garg, V. V. Vazirani, and M. Yannakakis, "Primal-dual approximation algorithms for integral flow and multicut in trees," *Algorithmica*, vol. 18, pp. 3–20, 1997.

[17] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, 1979.

[18] P. Tseng and D. P. Bertsekas, "Relaxation methods for problems with strictly convex separable costs and linear constraints," *Mathematical Programming*, vol. 38, pp. 303–321, 1987.

[19] *MATLAB User Manual*. Simulink Corp.