

Traffic Grooming Under a Sliding Scheduled Traffic Model in WDM Optical Networks

Bin Wang, Tianjian Li, Xubin Luo, Yuqi Fan
Department of Computer Science and Engineering
Wright State University, Dayton, OH 45435 USA
bwang@cs.wright.edu

Abstract

Work on traffic grooming in mesh networks considered several traffic models: static traffic model, dynamic random traffic model, incremental traffic model, and traffic matrix set model. While these different traffic models are valid and useful in many circumstances, they are not able to capture the traffic characteristics of applications that require capacity during specific time intervals. In this work, we argue for a general scheduled traffic model called sliding scheduled traffic model. In this model, the setup time t_s of a demand whose holding time is τ time units is not known in advance. Rather t_s is allowed to begin in a pre-specified time window $[\ell, r]$ subject to the constraint that $\ell \leq t_s \leq r - \tau$. The model is useful for service provisioning to many applications when fully dynamic speedy provisioning is still not widely available or deployed at least in the near future. We then propose efficient traffic grooming algorithms (time window based algorithm and traffic matrix based algorithm) under such a traffic model. Simulation results and comparison with a customized tabu search scheme show that the proposed algorithms are effective in meeting demand specifications and reducing total network resources.

1. Introduction

Applications differ considerably in their bandwidth requirements. For example, command and control systems might require a smaller amount of bandwidth while a high-resolution video conferencing application might call for a much larger amount of bandwidth. Many applications require much less bandwidth than a full wavelength can offer. Occupying a full wavelength for a few megabytes of data results in very poor utilization of network resources. On the other hand, wavelength channels operate at the peak electronic speed, making it extremely expensive to electronically process traffic on all wavelengths and fibers going

through a switch. Therefore, wavelength division multiplexing (WDM) systems employ optical add/drop multiplexers (OADMs) which allow a wavelength to either be dropped at a node or optically bypass the node's electronics. Traffic grooming [1] is construed as a multiplexing mechanism by which low-rate traffic streams can be appropriately aggregated and assigned to wavelength channels with the objective of efficiently utilizing network resources and minimizing the cost of electronic processing in the network.

Much of today's physical layer network infrastructure (e.g., metro networks) has been built with ring topologies using ADMs to add/drop wavelengths and/or tributary circuits. For example, ADMs, potentially one for each wavelength at each node in SONET/WDM networks, are capable of aggregating lower-rate SONET signals into a single higher-rate SONET stream. Much previous work on traffic grooming has been done in simple networks (e.g., ring, star, torus) [1]. The goal was to minimize the number of ADMs, or in addition to minimization of number of ADMs, to minimize the number of wavelengths used given (1) a static set of traffic demands consisting of low-rate circuits between source and destination node pairs; or (2) a dynamically changing traffic model.

Current and future WDM networks are increasingly arranged in general mesh topologies. Indeed, much recent work has focused on grooming traffic in mesh networks, see [2–17] for examples. Existing work on traffic grooming has considered several types of traffic model: static traffic model, dynamic random traffic model, incremental traffic model, and traffic matrix set model. In the static traffic model, all capacity demands are known in advance and do not change over time. For instance, a client company may request virtual private network capacity for connectivity among different company sites from a service provider. The objective is typically to minimize the network resources needed, e.g., the amount of line terminating equipment (LTE), e.g., ADMs, or, in general, the number of ports (see [10, 12, 14, 17] for examples), or to maximize the network throughput given a resource constraint (see [7] for exam-

ple). This model does not allow dynamic call setup and tear-down. In the dynamic random traffic model, a demand is assumed to arrive at a random time and last for a random amount of time. Usually statistical models are used. These models assume certain arrival statistics (e.g., Poisson process) and lasting time (e.g., exponential distribution) for demands, as well as a certain traffic distribution (e.g., uniform traffic). The design objective is typically to minimize the percentage of blocked traffic (see [6] for examples). Other traffic models have been considered for network planning and configuration. The work in [18] on multi-period network planning was based on an incremental traffic model and conducted network planning across several years to producing incrementally a network capable of carrying all traffic predicted up to the end of the planning horizon. The work reported in [19] considered time-variant offered traffic in the form of a set of traffic matrices at different instants for off-line configuration so as to accommodate such time-varying traffic. The work in [2] also used a set of traffic matrices to design and dimension a WDM mesh network to groom dynamically varying traffic.

While these different traffic models are valid and useful in many circumstances, these models are not able to capture the traffic characteristics of applications that require capacity during specific time intervals. For instance, a client company may request some scheduled demands for bandwidth from a service provider to satisfy its communication requirements at a specific time, e.g., between headquarters and production centers during office hours or between data centers during the night when backup of databases is performed and so on. Other examples include many US Department of Energy large-scale science applications (e.g., applications in high energy physics, climate data and computations, astrophysics, etc) that must deliver, at scheduled time durations, hundreds of Gbps throughput between two applications in near future and several Tbps within the next decade, ranging from cooperative remote visualization of massive archival data through the distribution of large amount of simulation data, to the interactive evolution of computations through computational steering [20]. These applications require provisioning of scheduled dedicated channels or sub-wavelength bandwidth pipes at a specific time with certain duration. These scheduled capacity demands are dynamic in nature. They are not static in the sense that the demands only last during the specified intervals. They are not entirely random either.

In this work, we argue for such a scheduled traffic model called *sliding scheduled traffic model*. In this model, the starting time t_s of a demand whose lasting time is τ is not known in advance. Rather t_s is allowed to begin in a time window $[\ell, r]$ subject to the constraint that $\ell \leq t_s \leq r - \tau$. The details of this model is described later in Section 2. The model is useful especially given that many applications ex-

hibit the aforementioned capacity requirement characteristics and services need to be provided when fully dynamic speedy provisioning is still not available or deployed at least in the near future. We shall assume no wavelength conversion. Without optical wavelength conversion, routing of a session is subjected to the wavelength continuity constraint which dictates that light-paths corresponding to a given session must travel on the same wavelength on all links from the source node to the destination node. Moreover, we shall assume that no traffic bifurcation is allowed. We use a demand time conflict reduction algorithm proposed in [21] to properly place scheduled demands within their respective time windows to minimize overlapping among demands in the time domain. We then propose efficient traffic grooming algorithms (time window based algorithm and traffic matrix based algorithm). In addition, we consider demand priority in the traffic grooming problem and, in case that a demand is blocked, how to rearrange the demand by negotiating a new setup time. We do not address the signaling aspect in this work however. To the best of our knowledge, traffic grooming under the sliding scheduled traffic model has not been studied in the literature. Our performance evaluation shows that the proposed traffic grooming algorithms are effective in meeting demand specifications and reducing total network resources used under the proposed traffic model.

The rest of the paper is organized as follows. Section 2 describes a new traffic model, sliding scheduled traffic model. Section 3 states the space-time traffic grooming problem and our proposed traffic grooming algorithms. Simulation results are reported in Section 4. The paper concludes in Section 5.

2. Sliding Scheduled Traffic Model

In this section, we describe a general scheduled traffic model called *sliding scheduled traffic model* proposed in [21]. We believe that this model is viable for many practical applications as discussed in Section 1.

Given a network topology $G = (N, L)$, where N is the set of nodes and L is the set of links. Each link has W wavelengths. A set of capacity demands M is given, each of which is represented by a tuple $(s, t, n, \ell, r, \tau, p)$ (Fig. 1(a)) that satisfies $r - \ell \geq \tau > 0$ where s and t are the source and destination, n is the number of requested capacity units, ℓ and r are the starting time and ending time of a time window during which the demand that lasts for τ time units resides, and p indicates the priority of the demand ($p = 1$ indicates a high priority and $p = 0$ a low priority). In this model, the demand lasting time τ is an interval within a time window $[\ell, r]$. Rather than fixing the starting time and ending time of the demand, we introduce a flexibility in the definition of the interval. As a result, the demand is allowed

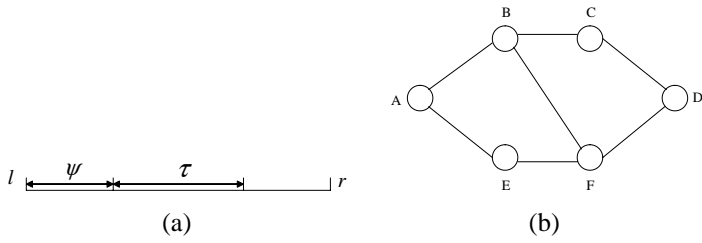


Figure 1. (a) The sliding scheduled traffic model; (b) An example network.

to slide within a larger time window $[\ell, r]$. This model allows an application to specify a larger time window during which the demand for communication capacity is movable and needs to be satisfied. Fixing the starting time and ending time of a demand may be too restrictive in practical scenarios. Furthermore, this model gives a service provider more flexibility in provisioning the requested demand and a better opportunity to optimize the network resources since a demand is considered accommodated as long as it is provisioned within the larger time window. Given a demand $d = (s, t, n, \ell, r, \tau, p)$, the actual starting time of the demand is variable relative to the left boundary ℓ of its associated time window. If the demand starts at ψ time units after ℓ , the demand is active during $[\ell + \psi, \ell + \psi + \tau]$. We represent this placement of the demand interval within the time window by (d, ψ) or $(s, t, n, \ell, r, \tau, p, \psi)$.

A special case of the model is when $\tau_i = r_i - \ell_i$ for all $d_i \in M$, in which case the starting time and ending time of demands are the same as the starting time and ending time of their respective time windows. A demand d_i can then be represented as $(s_i, t_i, n_i, \alpha_i, \beta_i, p_i)$ where s_i and t_i are the source and destination, n_i is the requested capacity units, α_i and β_i are the demand starting time and ending time, and p_i is the priority. An example scheduled demand set is given in Table 1 that has seven demands in an example network shown in Fig. 1(b).

The sliding scheduled demand traffic model is different from the static and dynamic random traffic models generally assumed in the literature. Static traffic means that all the demands are known in advance and do not change over time, whereas dynamic random traffic assumes that the inter-arrival time and lasting time of demands are random or conform to some probability distribution. The traffic model that is closest to ours is perhaps that of [22, 23] where a scheduled light-path demand model was considered. The routing and wavelength assignment problem is then solved using a branch & bound algorithm and a tabu search algorithm. Traffic grooming was not considered their work. The work in [23] considered scheduling of periodic connections with time flexibility on a single WDM link. Subramaniam

Time Window Division Algorithm (M)

```

//  $M$ : the set of demands to be scheduled
 $T$  = the set of possible division points of  $M$ ;
 $t_0 = 0$ ;  $t_j = T_j$ ;  $D = \text{empty}$ ;
for  $i = 1$  to  $|T|$ 
   $t_2 = T_i$ ;
  find all demands in  $[t_0, t_2]$  and put them in  $D$ ;
  find the cardinality of the maximum independent set of  $D$ ,  $D^*$ ;
  if ( $D^* > 1$ ) then
    a new time window  $TW[t_0, t_i]$  is determined and put all
    demands in  $[t_0, t_i]$  in the time window;
     $t_0 = t_i$ ;  $D = \text{empty}$ ;
  endif
   $t_1 = t_2$ ;
endfor
create the last time window  $TW[t_0, t_j]$  that contains all demands in  $[t_0, t_j]$ ;

```

Figure 2. Pseudo code for time window division algorithm.

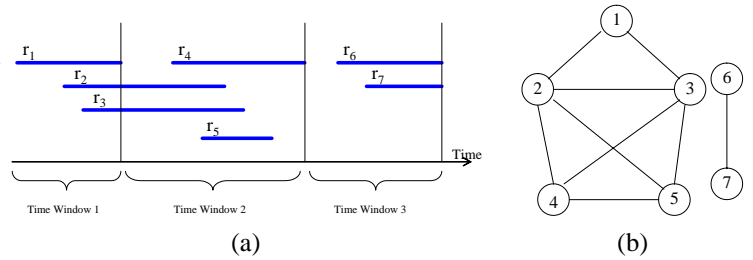


Figure 3. (a) A time window division example. (b) An interval graph representation of scheduled demands.

et al. [24] dealt with scheduling of multirate static sessions in time division multiplexed wavelength-routing ring networks. Our sliding scheduled traffic model is more general and dynamic in nature since it takes into account the evolution of the traffic load in the network over time, i.e., the time dimension of demands is explicitly considered since many capacity demands in ultra high-speed networks will be short-lived in contrast to 7×24 operations. We believe that this model is more suitable to characterize the resource requirements of certain network applications. Note that the model may allow the applications to negotiate their starting time with a service provider. This flexibility introduces problems that need to be solved for traffic grooming: placement of demand time intervals within their time windows, resource conflict/reuse in the spatial and/or temporal domains.

3. Proposed Traffic Grooming Algorithms

3.1. Space-Time Traffic Grooming Problem

Given a set of sliding scheduled traffic demands M , the problem is to (i) properly place each demand interval within its corresponding time window such that overlapping among demands in the time domain is minimized; (ii) route and groom demands such that

- In the **non-blocking case** in which the network has enough resources to accommodate all the demands in M to meet their specifications (i.e., capacity requirement and schedule requirement), the goal is to minimize the total network resources used in terms of, for example,
 1. the total number of wavelength-links used by all demands; or
 2. the maximum number of wavelengths needed on a link.
- In the **blocking case** in which the network does not have enough resources to accommodate all the demands as specified, in addition to minimization of total network resources used, the goal is, for example,
 1. to minimize the number of demands to be rearranged (i.e., to minimize the subset of demands that may have their schedule changed: postpone or prone the demands) in order to have all the demands in the set M accommodated by the network; or
 2. to minimize the total time from the time when the first demand starts to the time when the last demand ends (termed as the *schedule length*) it takes to satisfy all the demands in M .

In this work, our primary objective of traffic grooming is to minimize total wavelength-links used while trying to meet demands' specifications. The hardness of the problem lies in the spatial and temporal constraints imposed on the set of demands. In the spatial domain, demands are routed through the mesh network topology and may share the same wavelength on the same link. Grooming of demands is also subjected to the wavelength continuity constraint since there is no wavelength conversion. In the time domain, demands may overlap in time. In the performance evaluation, we use a demand time correlation factor [22] to characterize the extent of conflicts among demands in the time domain. The problem is therefore termed as space-time traffic grooming problem. The problem can be easily shown to be NP-hard from reduction of the multiprocessor-scheduling problem [25]. In the following, we propose heuristic traffic grooming algorithms that take advantage of resource reuse in both domains.

3.2. Demand Placement in a Time Window

Obviously, reducing overlapping or conflict between demands in time helps temporal resource reuse. Given a set of demands $M = \{d_i \mid 1 \leq i \leq m\}$, we proposed a time domain conflict reduction algorithm in [21] applied to the demand set M to find a proper placement of demand intervals in their associated time windows (i.e., find a set of $\psi_i, 1 \leq i \leq m$ for each demand $d_i \in M$) such that the number of demand pairs (regardless of their potential spatial routing) that overlap (or conflict) in time is minimized. This algorithm will be used in this work. In what follows, we consider a set of demands that have been properly placed using the proposed time conflict reduction algorithm.

3.3. Time Window Based Grooming Algorithm

Our algorithm is based on the following observations. Demands that overlap in time must be disjoint in the spatial domain if the total capacity of these demands exceeds the wavelength capacity (e.g., these demands cannot all share the same wavelength on the same link due to wavelength capacity limit). Network resources used by one demand can be reused by another demand as long as they do not overlap in time. This motivates us to divide the set of demands into subsets based on the demands' starting time and ending time such that demands in different subsets are disjoint in time.

We model the time conflicts of the set of demands M (e.g., Fig. 3(a)) using an interval graph (Fig. 3(b)). Based on the interval graph representation, we can then divide the demands into subsets called time windows. *Note that these time windows are not related to the "time window" of a demand in the sliding scheduled traffic model.* An algorithm that divides demands into time windows is given later. A virtual network topology is associated with each time window. Note that it is not always possible to assign a demand into a single time window. This demand is termed as a straddling demand, e.g., r_2 and r_3 of Fig. 3(a). The resources used by demands in one time window can potentially be reused by other demands in other windows. For straddling demands the same resources have to be reserved across time windows. In the following, we first describe the details of dividing the demand set into time windows and the algorithm for routing and grooming demands based on these windows. We then give the algorithm for the space-time traffic grooming problem.

Time Window Division The algorithm divides the set of demands into disjoint time windows. Demands within a time window overlap in time *pairwise*. Therefore, demands within the same window have to be disjoint in the spatial domain subject to the wavelength capacity limit, e.g., if the total capacity of demands exceeds the wavelength capac-

demand	s	d	n	α	β	p
r_1	B	F	1	05:00	09:20	0 (low)
r_2	B	D	1	07:00	12:40	0 (low)
r_3	A	C	2	08:00	14:00	1 (high)
r_4	A	D	2	11:00	16:00	0 (low)
r_5	E	D	3	12:00	14:50	0 (low)
r_6	C	E	1	17:00	21:00	1 (high)
r_7	F	A	3	18:00	21:00	0 (low)

Table 1. An example of a scheduled demand set.

ity, some demands cannot share the same wavelength on the same link. Specifically, let

$$T = \cup_{i=1}^{|M|} \{\beta_i\} \quad (1)$$

be an ordered set of $|T|$ demand ending time values $T_1 < T_2 < \dots < T_{|T|}$ ($|T| \leq |M|$ since some β_i 's may be the same). We take the values in T as possible division points. We define that a demand lies in a time interval if its starting time or ending time lies in the interval. We then adapt the maximum independent set algorithm over an interval graph [26] to find a maximum time interval (starting from the ending time of the previous time window) during which all demands overlap pairwise in time. The algorithm pseudo code is given in Fig. 2. Fig. 3(a) shows the time window division result of an example demand set shown in Table 1. We can relax the ‘‘pairwise overlapping in time’’ requirement by allowing a tunable parameter (not shown in the code) that specifies the number of demand pairs that are disjoint in time.

After the time window division, a demand may reside in a time window or straddle two or more windows. In the latter case, the demand is accommodated only when the same path can be found and resources on the path can be reserved in all the virtual networks associated with all the time windows it straddles. Otherwise the demand has to be rearranged by negotiating a new starting time (i.e., postpone or propone the demand) so that sufficient network resources can be provided in the new windows it straddles. Note that existing demands are not rerouted and no traffic bifurcation is allowed.

Routing and Grooming Based on Time Windows After dividing the demand set M using the time window division algorithm, we have a number of demand subsets: $D^{SH}, D^{SL}, TW_i^H, TW_i^L$ where D^{SH} and D^{SL} are the sets of high-priority straddling demands and low-priority straddling demands, respectively; TW_i^H and TW_i^L are the sets of high-priority and low-priority demands within time window i , respectively. Table 2 shows the subsets after the time window division of the example shown in Table 1.

We keep a virtual network G_i for each time window TW_i to record its residual network resources in the physical network topology. Moreover, we maintain a virtual link topology Q_i for each time window to keep track of the residual resources on the virtual links. The virtual link topol-

	Window1	Window2	Window3
TW^H	\emptyset	\emptyset	r_6
TW^L	r_1	r_4, r_5	r_7
D^{SH}	r_3	r_3	\emptyset
D^{SL}	r_2	r_2	\emptyset

Table 2. Subsets after time window division.

ogy is updated as light-paths are established and demands are accommodated. Initially, there is no node and link in the virtual link topology. Once a demand between (s, d) is accommodated and a light-path is established for the demand in the time windows it straddles, we put a virtual link (s, d) into the virtual link topology where s and d are the source and destination nodes of the demand. In addition, the remaining capacity units on the light-path are assigned to the virtual link as its available capacity units. For each non-straddling demand $\in TW_i$, we apply a modified Dijkstra’s algorithm (Fig. 5) to the wavelength graph of the combined network $(G_i \cup Q_i)$ to find the best path for the demand in time window TW_i . By best path, we mean that the number of capacity units available on all the links (physical as well as virtual) along the path satisfies the capacity requirement of the demand and the path has the minimum total cost. Link cost assignment is given in Fig. 5. Once such a path is found for the demand, we update the network state information. If the demand cannot be satisfied in time window TW_i , the next procedure depends on the priority of the demand. If its priority is high (i.e., the demand belongs to some TW_i^H), this demand can be demoted to TW_i^L and given a preferential treatment within TW_i^L ; otherwise (i.e., the demand belongs to TW_i^L), we put the demand into demand subset D^R in which all the demands need to be rearranged (Fig. 4). For each demand subset, we select a demand to route and groom in the descending order of request capacity in the set. Our objective is to accommodate as many demands with higher capacity requirements as possible first. Therefore, this algorithm, in this sense, is a greedy algorithm.

For straddling demands (in D^{SH} or D^{SL}), we handle them in the same way except that the network in which each demand is routed is obtained by combining G_i ’s and Q_i ’s over all the time windows straddled by the demand, and all these networks must be updated if a path is found. The pseudo code of the algorithm that handles TW_i^H, TW_i^L, D^{SH} , and D^{SL} is given in Fig. 4.

Demand Rearrangement and Grooming Algorithm If network resources are not sufficient to meet all the demands’ specifications (including schedule requirement) given in their original requests, some demands have to be rearranged. That is, these demands can be accommodated at the cost of changed schedules. D^R is the demand set in which all the demands must be rearranged. For each demand in D^R , the algorithm tries to find one or

```

Greedy Time Window Grooming Algorithm ( $G, D$ ) //  $D$ : set of demands
while ( $D$  not empty) do
  choose demand  $r$  which has the highest capacity requirement in  $D$ ;
   $G' = G$ ;
  for each time window  $TW_i$  straddled by demand  $r$ 
     $G' = G' \cap (G_i \cup Q_i)$ ; //  $Q_i$ : virtual link topology associated with  $TW_i$ 
  endfor
  run Routing Algorithm ( $G', r$ ) to find the best path  $p_r$  for demand  $r$  in  $G'$ ;
  if  $p_r$  is found
     $D = D - r$ ;
    for each time window  $TW_i$  straddled by demand  $r$ 
      update  $G_i, Q_i$ ;
    endfor
  else
    put  $r$  in  $D^R$ 
  endif
endwhile

```

Figure 4. Greedy time window based grooming algorithm.

```

Routing Algorithm ( $G, r$ )
 $u$  = capacity requirement of demand  $r$ ;
for each wavelength  $\lambda$ 
  for each virtual link  $l$  (corresponding to an established path  $p$ ) in  $G$ 
     $b_\lambda$  = minimum free capacity units of wavelength  $\lambda$  on all the physical links along  $p$ ;
    if ( $b_\lambda < u$ )
       $cost(l, \lambda) = \infty$ ;
    else
       $cost(l, \lambda) = Length(p)$ ; //length of the path
    endif
  endfor
  for each physical link  $l$  in  $G$ 
     $b_\lambda$  = minimum free capacity units of wavelength  $\lambda$  on  $l$ ;
    if ( $b_\lambda < u$ )
       $cost(l, \lambda) = \infty$ ;
    else if wavelength-link ( $l, \lambda$ ) has been used by some established connections
       $cost(l, \lambda) = Length(l)$ ; //length of the link
    else
       $cost(l, \lambda) = M + Length(l)$ ; //  $M$ : a big number
    endif
  endfor
  endfor
   $G_\lambda = G$  with above link cost assignment;
   $p_\lambda$  = the least cost path for demand  $r$  in  $G_\lambda$ ;
endfor
return  $p^*$  = the least cost path among all  $p_\lambda$ 's;

```

Figure 5. Basic routing algorithm.

more time windows in which the demand can be accommodated and the window(s) starts/start as early as possible. The rationale is that we prefer to route and groom the demand by reusing resources and we would like to prevent the rearrangement of the demand from prolonging the total time it takes to schedule all demands in M (also termed as the schedule length). The pseudo code of the algorithm for routing and grooming rearranged demands is given in Fig. 6. Other demand rearrangement policies can also be designed.

Space-Time Traffic Grooming Algorithm Given the space-time traffic grooming problem, our objective is to accommodate as many demands as possible with both their

```

Rearrange Grooming Algorithm ( $G, D$ ) //  $D$ : set of demands
while ( $D$  not empty) do
  choose demand  $r$  which has the highest capacity requirement in  $D$ ;
   $p_r = empty$ ;  $i = 1$ ; //  $p_r$ : route for demand  $r$ 
  while ( $p_r = empty$ ) do
    find smallest  $j$  ( $i \leq j$ ) such that windows  $i$  to  $j$  can accommodate  $r$ ;
    if no such time window is found
      create a new time window  $TW_j$  with  $G_j$  for demand  $r$ ;
    else
       $G' = (G_i \cup Q_i) \cap \dots \cap (G_j \cup Q_j)$ ;
    endif
    run Routing Algorithm ( $G', r$ ) to find the best path  $p_r$  for demand  $r$  in  $G'$ ;
    if  $p_r$  is found
       $D = D - r$ ;
      for each time window  $TW_i$  straddled by demand  $r$ 
        update  $G_i, Q_i$ ;
      endfor
    else
       $i = i + 1$ ;
    endif
  endwhile
endwhile

```

Figure 6. Demand rearrangement and grooming algorithm.

```

Space Time Traffic Grooming Algorithm ( $G, M$ )
run Time Window Division Algorithm ( $M$ );
run Greedy Time Window Grooming Algorithm ( $G, D^{SH}$ );
run Greedy Time Window Grooming Algorithm ( $G, TW_i^H$ ) for all  $TW_i$ ;
run Greedy Time Window Grooming Algorithm ( $G, D^{SL}$ );
run Greedy Time Window Grooming Algorithm ( $G, TW_i^L$ ) for all  $TW_i$ ;
if ( $D^R$  not empty), run Rearrange RWA Algorithm ( $D^R$ );

```

Figure 7. Space-time traffic grooming algorithm.

capacity requirements and schedule requirements. If not all the demands can be satisfied in both space and time domains at the same time, some demands must be selected as victims to be rearranged. Therefore, to route and groom the demand set M , we try to route and groom high-priority demands first, and then route and groom low-priority demands. In addition, straddling demands are routed and groomed prior to the demands residing in a single time window since it is more likely to route a straddling demand earlier. The pseudo code of the overall time window based space-time traffic grooming algorithm is given in Fig. 7.

4. Performance Evaluation

We use the NSFNET topology in our performance evaluation and comparison. A weight that represents the physical length is associated with each link of the topology [22]. Each link has 30 wavelengths unless specified otherwise. Different demand sets containing 50 to 400 capacity demands are used in the simulation. We consider three classes

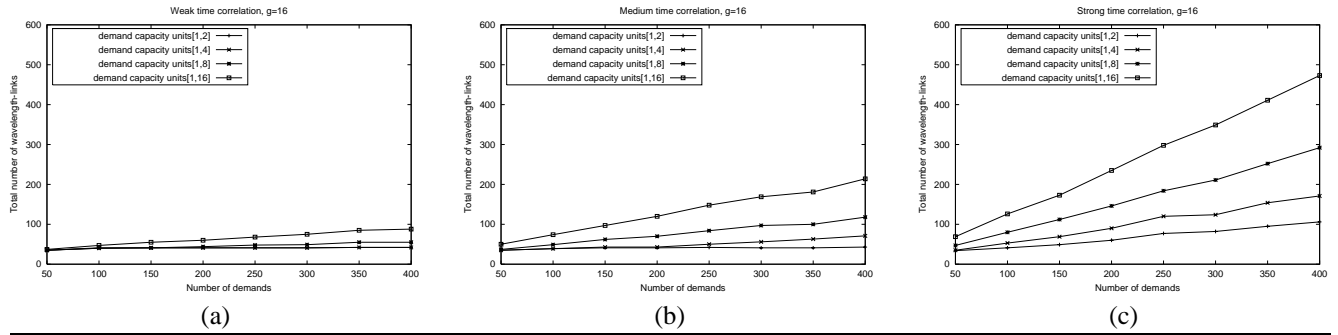


Figure 8. Time window based grooming algorithm: Total number of wavelength-links used with grooming factor $g = 16$ versus number of demands when (a) demand time correlation is weak; (b) demand time correlation is medium; (c) demand time correlation is strong.

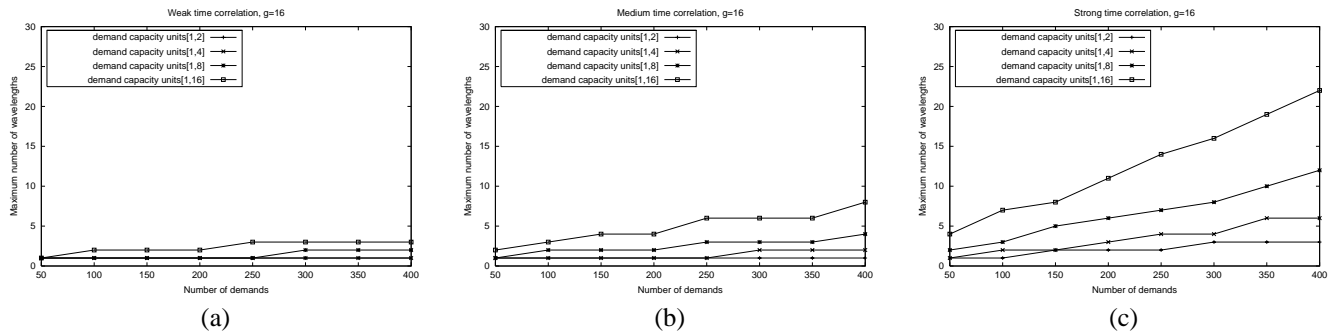


Figure 9. Time window based grooming algorithm: Maximum number of wavelengths used on a link with grooming factor $g = 16$ versus number of demands when (a) demand time correlation is weak; (b) demand time correlation is medium; (c) demand time correlation is strong.

of demand set with a demand time correlation factor being weak (0.01), medium (0.5), and strong (0.8) to characterize the conflicts among demands in the time domain. The source and destination of a demand are randomly generated. The grooming factor g varies from 4 to 32. Given a grooming factor g , the number of capacity units of a demand is drawn from a uniform distribution in the intervals $[1,2], [1,4], [1,8], \dots, [1,g]$, respectively for different cases. Y is set to 1 in the traffic matrix based grooming algorithm. In all the cases simulated, no demand was blocked. Priority of all demands is set to the same. Simulation was run to 95% confidence.

We are interested in the following performance metrics: (i) the total number of wavelength-links used in the entire network; (ii) the maximum of wavelengths used on a link. The primary objective of our algorithms is to route and groom the set of traffic demands so that the total number of wavelength-links used in the entire network is minimized when network resources are sufficient to meet de-

mand specifications.

4.1. Time Window Based Grooming Algorithm

Fig. 9 shows the maximum number of wavelengths used on a link with grooming factor $g = 16$ versus number of demands for three different cases of demand time correlation. Fig. 8 depicts the total number of wavelength-links used in the network with grooming factor $g = 16$ versus number of demands for three different cases of demand time correlation. From the figures (Fig. 9 and 8), we observe that (1) The total number of wavelength-links used and the maximum number of wavelengths needed on a link increase when the number of demands increases. (2) The total wavelength-links used and the maximum number of wavelengths on a link increase as the average demand capacity increases for a fixed number of demands. The speed of increase picks up as the average demand capacity gets larger. This shows that when the average demand capacity is small

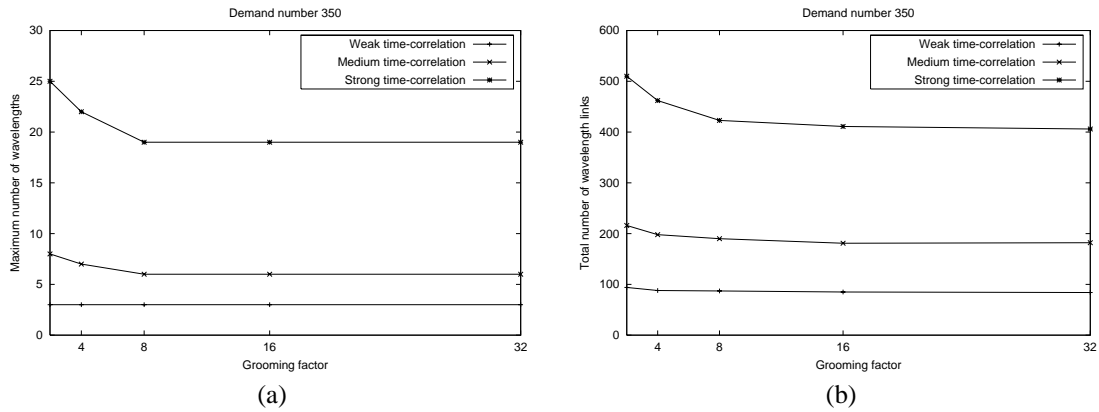


Figure 10. Time window based grooming algorithm: (a) Maximum number of wavelengths used on a link; (b) Total number of wavelength-links used, versus grooming factor g for different demand time correlation when the number of demands is 350 and the capacity of a demand is drawn from $[1, g]$.

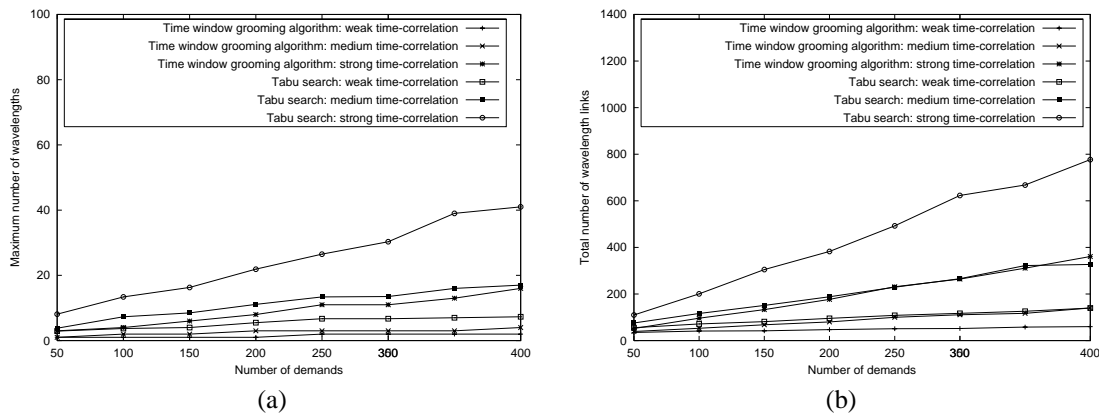


Figure 11. Comparison of time window based grooming algorithm with tabu search, grooming factor $g = 4$, demand capacity is drawn from $[1, 2]$, (a) Maximum number of wavelengths used on a link; (b) Total number of wavelength-links used, versus number of demands when demand time correlation is weak, medium, and strong.

relative to the grooming factor, traffic grooming is more effective. Moreover, demand time correlation plays an important role in traffic grooming and has a significant impact on the cost of a traffic grooming algorithm. A stronger demand time correlation factor translates to a less amount of traffic grooming gain. A set of demands with significant time disjointness should ease the reuse of wavelength-links and as a result, lead to a smaller number of required wavelength-links. (3) When the network load is relatively low, the network resources needed show only a slight increase as the average demand capacity increases, especially when the demand time correlation is medium or weak. This indicates that when the traffic load is light, traffic grooming effec-

tively packs demands onto a small set of wavelengths, reducing network resources needed.

Fig. 10(a) and (b) shows the maximum number of wavelengths used on a link in the network and the total number of wavelength-links used versus grooming factor g for demand sets with different demand time correlation when the number of demands is 350 and the capacity of a demand is uniformly drawn from $[1, g]$. As seen from the figures, when the grooming factor g increases, the amount of resources used in the network decreases and levels off when g becomes relatively large. The decrease in the amount of resources used is more significant when the demand time correlation is stronger. This shows that a finer grooming granularity re-

sults in a larger grooming gain and the traffic grooming algorithm is effective in reducing network resources used for demand sets with strong time correlation.

4.2. Comparison with Tabu Search

Since no previous work considered our traffic model, we compare the performance of traffic matrix based grooming algorithm against that of solutions obtained using a customized tabu search scheme that tries to minimize wavelength-links used. For fair comparison, the number of wavelengths is set to be not a limiting factor. We consider three classes of demand set with a demand time correlation factor being weak (0.01), medium (0.5), and strong (0.8).

Tabu search is an iterative meta-heuristic algorithm used for combinatorial optimization problems. The algorithm explores the solution space (e.g., let x denote a solution and x_i the x_i -th alternate path for demand i in our problem) until either a number of iterations is reached or a specific cost criterion is satisfied. The exploration starts with an initial solution (e.g., a random solution). This solution is the current solution at the beginning of the algorithm. At each iteration, the algorithm computes a set of neighboring solutions. Neighboring solutions are solutions that can be reached from the current solution by perturbation applied to the current solution. The solution in the neighborhood with the best cost is selected as the new current solution. To prevent the algorithm from cycling through the same set of current solutions, a tabu list is maintained. The solution selected at each iteration as the current solution is inserted in the list and stays there for a given number of iterations. The list keeps track of a number of previously explored solutions and prohibits tabu search from revisiting them again as long as they are on the list. In this way, tabu search can overcome local minima by forcing the acceptance of solutions worse than the current solution. In our customized tabu search, the size of neighborhood is 100 and the length of tabu list is 2000. The algorithm creates a new solution from a current one by randomly selecting a demand from the demand set M and randomly modifying the selected route for this demand. Specifically, given a set of demands M , the algorithm uses the following steps to generate a neighbor: (1) generate a random number $i \in [1 \dots |M|]$, where $|M|$ is the number of demands; (2) generate a random number $j \in [1 \dots K_{MAX}]$, where K_{MAX} is the number of alternate paths for each demand (in our simulation, K_{MAX} is 4); (3) Replace $x_i, 1 \leq i \leq |M|$ by j , where j is the j -th alternate path of demand i . In addition, our tabu search performs diversification by applying multiple perturbations (perturbation number is 4 in our case) to the current solution if no cost improvement has been obtained after a large number (20 in our case) of iterations. Diversification guides

tabu search towards unexplored parts of the search space. In this way the solution space will be covered more thoroughly and the chance of missing the optimal solution will be reduced. The above steps are repeated until the required number of neighbors are generated or a maximum tabu search iterations (1000 in our case) is reached. We use tabu search to provide solutions with a cost “close” to the optimal one.

Fig. 11 compares the performance of time window based grooming algorithm against that of a customized tabu search scheme when the grooming factor $g = 4$, demand capacity is drawn uniformly from [1, 2], and demand time correlation is weak, medium, and strong, respectively. From the figure, we observe that the performance of the proposed algorithm outperforms the customized tabu search scheme in all cases. The reason is that our customized tabu search scheme uses fixed alternate routing for each demand and the number of candidate paths is limited ($K_{MAX} = 4$). Time window based grooming algorithm, however, uses a modified Dijkstra’s algorithm to dynamically find a path for each demand, which does a better job of packing wavelength channels and reusing resources. These comparisons show the effectiveness of the proposed algorithms in meeting demand specifications and reducing network resources used.

5. Conclusions

In this work, we argue for a general *sliding scheduled traffic model* to capture the traffic characteristics of applications that require capacity during specific time intervals. The model is useful for service provisioning to many applications when fully dynamic speedy provisioning is still not widely available or deployed at least in the near future. We have proposed efficient traffic grooming algorithms (time window based algorithm and traffic matrix based algorithm) for scheduled capacity demands. Our simulation results and comparison with a customized tabu search scheme have shown that the proposed traffic grooming algorithms are effective in meeting demand specifications and reducing total network resources used.

References

- [1] E. Modiano and P. J. Lin, “Traffic grooming in WDM networks,” *IEEE Communication Magazine*, pp. 124–129, July 2001.
- [2] N. Srinivas and C. S. R. Murthy, “Design and dimensioning of a WDM mesh network to groom dynamically varying traffic,” *Photonic Network Communications*, vol. 7, no. 2, pp. 179–191, 2004.
- [3] J. Fang and A. K. Somani, “Enabling subwavelength level traffic grooming in survivable WDM optical network design,”

- Proceedings of IEEE Globecom, San Francisco CA USA*, December 2003.
- [4] H. Zhu, K. Zhu, H. Zang, and B. Mukherjee, "Cost-effective WDM backbone network design with OXCs of different bandwidth granularities," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 9, pp. 1452–1466, November 2003.
 - [5] H. Zhu, H. Zang, K. Zhu, and B. Mukherjee, "A novel generic graph model for traffic grooming in heterogeneous WDM mesh networks," *IEEE/ACM Transactions on Networking*, vol. 11, no. 2, pp. 285–299, April 2003.
 - [6] S. Zhang and B. Ramamurthy, "Dynamic traffic grooming algorithms for reconfigurable SONET over WDM networks," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 7, pp. 1165–1172, September 2003.
 - [7] K. Zhu and B. Mukherjee, "Traffic grooming in an optical WDM mesh network," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 1, pp. 122–133, January 2002.
 - [8] C. Ou, K. Zhu, H. Zang, L. H. Sahasrabudh, and B. Mukherjee, "Traffic grooming for survivable WDM networks - shared protection," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 9, pp. 1367–1383, November 2003.
 - [9] A. E. Kamal and R. Ul-Mustafa, "Multicast traffic grooming in WDM networks," *Proceedings of OptiComm'03, Dallas TX USA*, pp. 25–37, October 2003.
 - [10] V. R. Konda and T. Y. Chow, "Algorithm for traffic grooming in optical networks to minimize the number of transceivers," *IEEE Workshop on High Performance Switching and Routing*, 2001.
 - [11] S. Thiagarajan and A. K. Somani, "Traffic grooming for survivable WDM mesh networks," *In Proceedings of OPTI-COMM*, 2001.
 - [12] R. Dutta and G. N. Rouskas, "Bounds on traffic grooming in star and tree networks," *Proceedings of the 2001 Allerton Conference on Communication, Control, and Computing*, October 2001.
 - [13] R. Srinivasan, "Dynamic routing in WDM grooming networks," *Iowa State University, Dept. of Electrical and Computer Engineering, DCNL Technical Report: DCNL-ON-2001-001*, 2001.
 - [14] M. Brunato and R. Battiti, "A multistart randomized greedy algorithm for traffic grooming on mesh logical topologies," *Proceedings of IFIP Working Conference on Optical Network Design and Modelling, Torino, Italy*, December 2002.
 - [15] R. Srinivasan and A.K. Somani, "Request-specific routing in WDM grooming networks," *Proceedings of IEEE International Conference on Communications (ICC 2002)*, April-May 2002.
 - [16] J. Q. Hu and B. Leida, "Traffic grooming, routing, and wavelength assignment in optical WDM mesh networks," *Sixth INFORMS Telecommunications Conference*, March 2002.
 - [17] R. Dutta and G. N. Rouskas, "Traffic grooming in WDM networks: Past and future," *NCSU CSC Technical Report TR-2002-02*, 2002.
 - [18] N. Geary, A. Antonopoulos, E. Drakopoulos, and J. O'Reilly, "Analysis of Optimisation Issues In Multi-Period DWDM Network Planning," *Proceedings of IEEE INFOCOM'01, Anchorage, Alaska USA*, April 2001.
 - [19] F. Ricciato, S. Salsano, a. Belmonte, and M. Listanti, "Off-line configuration of a MPLS over WDM network under time-varying offered traffic," *Proceedings of INFOCOM*, June 2002.
 - [20] N. S. Rao and W. R. Wing, "Network provisioning and protocols for DOE large-science applications," *Report of DOE workshop on ultra high-speed transport protocols and dynamic provisioning for large-scale science applications, Argonne, IL USA*, April 2003.
 - [21] B. Wang, T. Li, X. Luo, and Y. Fan, "Routing and wavelength assignment under a scheduled traffic model in reconfigurable WDM optical networks," *Manuscript submitted for publication*, 2004.
 - [22] J. Kuri, N. Puech, M. Gagnaire, E. Dotaro, and R. Douville, "Routing and wavelength assignments of scheduled lightpath demands," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 8, pp. 1231–1240, October 2003.
 - [23] W. Su and G. Sasaki, "Scheduling of periodic connections with flexibility," *41st Annual Allerton Conference on Communication, Control, and Computing, Monticello, IL USA*, October 2003.
 - [24] S. Subramaniam, E. J. Harder, and H. Choi, "Scheduling multirate sessions in time division multiplexed wavelength-routing networks," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 10, pp. 2105–2110, October 2000.
 - [25] M. Garey and D. Johnson, "Computers and Intractability: A guide to the theory of NP-Completeness," *W. H. Freeman, San Francisco*, 1979.
 - [26] U. I. Gupta, D. T. Lee, and J. Y. T. Leung, "Efficient algorithms for interval graphs and circular-arc graphs," *Networks*, pp. 459–467, 1982.