

Dynamic Scheduling of Network Resources with Advance Reservations in Optical Grids

Savera Tanwir¹, Lina Battestilli², Harry Perros³ and Gigi Karmous-Edwards²

¹ ISE Department
NUST Institute of Information Technology
Rawalpindi, Pakistan
E-mail: *savera.tanwir@niit.edu.pk*

² Advanced Technology Group
MCNC Grid Computing and Network Services
RTP, NC, USA
E-mail: {*lina, gigi*}@mcnc.org

³ Department of Computer Science
North Carolina State University
Raleigh, NC, USA
E-mail: *hp@csc.ncsu.edu*

August 19, 2007

Abstract

Advance reservation of lightpaths in Grid environments is necessary to guarantee QoS and reliability. In this paper, we have evaluated and compared several algorithms for dynamic scheduling of lightpaths using a flexible advance reservation model. The main aim is to find the best scheduling policy for a Grid network resource manager that improves network utilization and minimizes blocking. The scheduling of lightpaths involve both routing and wavelength assignment. Our simulation results show that minimum cost adaptive routing where link costs are determined by the current and future usage of the link provides the minimum blocking. For wavelength assignment, we have used a scheme that reduces fragmentation by minimizing unused gaps. We have also analyzed approaches for failure recovery and resource optimization.

1 Introduction

Optical Grids have emerged around the world to support the bandwidth requirements of many eScience and eBusiness applications. Some examples of such applications include Data Intensive applications such as DynaCode [1] for environmental modeling, Collaborative Remote Data Visualization [2] and High Definition Interactive Video Conferencing [3] that requires high speed transfer of high definition video streams with multicast capabilities. We have also recently seen great progress in the optical networks technology in terms of transmission capacity and dynamic reconfigurability. With Wavelength Division Multiplexing (WDM), a single wavelength can now reach OC-192 (10 Gbps) and hundreds of wavelengths can be supported on each fiber. So the Optical Grids have emerged to interconnect massive compute clusters and large storage resources through multi-gigabit lightpath connections. Although much progress has been made towards developing Grid technologies, the area that is still underdeveloped is the link between Grid applications and underlying network technologies which make Grids truly effective. The abstraction and encapsulation of these optical network resources into manageable and dynamically provisioned Grid entities is necessary in order to meet the complex demand patterns of Grid applications and to optimize the overall network utilization.

Various projects around the world consider the problem of building reconfigurable, dynamic, adaptable Optical Grids [4, 5, 6, 7]. These projects consider the network resources as key Grid resources that can be managed and controlled like any other Grid resource. They have developed

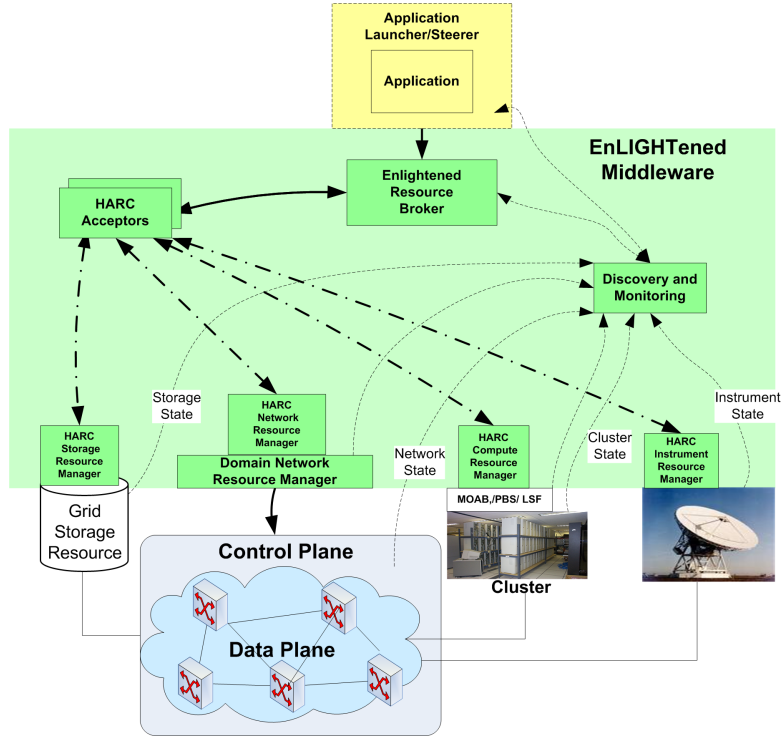


Figure 1: EnLIGHTened Middleware Architecture

Grid middleware that can make optimized use of the network as a virtual coordinated resource. Their Grid middleware is responsible for the resource managing, coordinating, allocating and scheduling advance resource reservations for all Grid resources: network, cluster, storage, scientific instruments, etc. The middleware establishes the optical connections with the necessary bandwidth and it interfaces with the cluster schedulers (e.g. Moab, OpenPBS/Torque, etc) to make advance reservations. From a Grid application perspective, all types of resources may be required to meet its computational needs.

The EnLIGHTened Computing middleware architecture is shown in detail in figure 1. Applications use the Application Launcher Steerer (ALS) to request resources from the EnLIGHTened Resource Broker (ERB). The request may include compute resources, network, storage, etc. The ERB then queries the underlying Discovery and Monitoring (DM) component for information, and then chooses the most appropriate resources. The Highly-Available Resource Co-allocator (HARC) Acceptors are used to co-allocate the required resources for the selected time range by using the HARC Resource Managers (RM). HARC either co-allocates *all* the needed resources or *none* at all by using a phased commit protocol. Its flexible design allows arbitrary resources to be co-allocated with advance reservations. The Compute RM manages the compute resources and wraps existing cluster schedulers (LSF, OpenPBS, Maui, etc) to provide the advance reservation capability. The HARC Network RM interfaces to the Domain Network RM (DNRM), which manages the network resources and supports advance reservations. The HARC Instrument RM interfaces to scientific instruments.

The DNRM is the middleware component that controls the optical network. It is responsible for path computation, wavelength assignment of the end-to-end lightpath and it keeps the advance reservation link/wavelength timetable. The DNRM discovers the network topology using control plane information such as OSPF-TE MIB information. It also monitors the network and provides up-to-date status of the network resources to the DM middleware component. The DNRM does the network resource allocation by interfacing with the specific control plane, i.e.,

each network domain may have a dynamic control mechanism such as DRAGON's GMPLS [8], CHEETAH [9], UCLP [10], etc. The DNRM implements mechanisms for restoration and recovery of lightpaths in case of failures. It also enforces policies and access control information. In a shared-lambda environment, the enforcement of policies is crucial for successful operation. Each DNRM has a policy engine that gives Grid administrators control over how networking resources are shared.

In an Optical Grid, application requests may be for *immediate* or for *advance* reservations. In the immediate reservations case, the application requests resources from the Grid middleware for immediate use. In the advance reservation case, the requests are for a future time, i.e., there is a time period between the request arrival and the time of the resource reservation. In fact, the immediate reservations can be viewed as a special case of the advance reservations, with a zero time period between the request arrival and the time of the resource reservation. We believe that in a shared Optical Grid, advance reservations are a necessity because they guarantee the availability of resources with the required QoS. For the cluster resources, certain schedulers, such as Maui [11], provide some advance scheduling capability. In this paper, we focus on the scheduling of advance reservations of network resources.

The concept of advance reservations in an optical network has attracted some attention in recent years. In [12] the design of effective RWA algorithms for different types of advance reservations is discussed and algorithms are presented for requests having specific start time and specific duration (STSD), specific start time and unspecified duration (STUD) and Unspecified Start Time and Specified Duration (UTSD). In [13], the author discussed the properties of advance reservation and proposed an architecture for a bandwidth broker to improve the performance of a network based on the additional knowledge of these future reservations. The Globus Architecture for Reservation and Allocation (GARA), presented in [14], supports advance reservations for various types of resources for the Grid. In [15], the authors also discussed advance reservation of heterogeneous network paths in the context of Grid computing and proposed a network resource hierarchy to integrate the path management with Grid information and authentication services.

In comparison to the immediate reservations, advance reservations generally degrade the resource utilization and the acceptance rate due to resource fragmentation [16]. This can be improved by introducing some flexibility in defining the advance reservations. In [17] the authors proposed a sliding scheduled traffic model and a demand time conflict resolution algorithm to maximize resource usage in a network. In [18] an ILP formulation for the RWA problem is proposed for shared and dedicated protection, assuming a predetermined static demand matrix. In [19] the authors proposed a Flexible Advance Reservation Model (FARM) and described how to implement this model in the meta-scheduling problem. The results indicate that using this approach the acceptance rate and resource utilization can be improved dramatically. We have also used the Flexible Advance Reservation Model in the lambda scheduler due to the mentioned advantages.

In this paper, our focus is to evaluate and compare various algorithms for advance lightpath scheduling that can be implemented in a DNRM. The main aim is to find the best scheduling policy for a Grid network resource manager that improves network utilization and minimizes blocking.

The rest of the paper is organized as follows: In Section 2 we describe the advance scheduling problem. In Section 3 we present the scheduler functions and algorithms. Section 4 gives the simulation model and the numerical results and we conclude in Section 5.

2 Problem Description

Consider a Network Topology Graph $G = (N, L, W)$ where N is the set of nodes, L is the set of links and W is the set of wavelengths supported by each link. A user submits an advance reservation request for a lightpath between any two nodes on G to the DNRM. Each request R is defined by the following parameters:

$$R = [\text{source node, destination node, } s, e, d, \text{ bandwidth}]$$

where d is the reservation duration, and s and e are the starting and ending time of the *scheduling window* respectively as shown in Figure 2. The time is slotted with a slot size equal to t' . The scheduling window defines the time period within which the requestor would like to make a resource reservation. The scheduling window must be bigger than the reservation duration d . Thus the scheduler must check if a path is available during interval $[s + t, s + t + d]$ where $t = 0, 1, 2, \dots, e - s - d$.

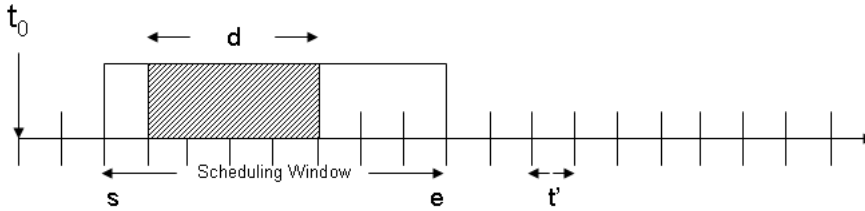


Figure 2: Scheduling Window

This is an online scheduling problem because the requests arrive dynamically and for each request R , the DNRM must compute a path and then check if a wavelength on each link of this path can be reserved for duration d within the scheduling window $[s, e]$. The DNRM allocates a wavelength on each link along a path from the source to the destination nodes. If a wavelength along the path for the specified period of time is not available, another path has to be determined. In order to do this, the DNRM maintains a schedule of the reservations called the *Reservation Table*. It contains all current and future reservations and it is used to search for available resources for new advance reservations.

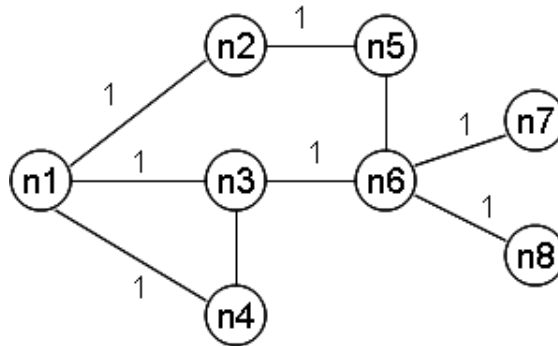


Figure 3: Example Topology

Table 1 shows an example of the reservation table for two lightpath requests for the network shown in figure 3. This is an optical network with 2 wavelengths per fiber link and each link has a cost of 1. Let us assume that at time t_0 two lightpath requests arrive, $R_1 = [n_1, n_7, t_1, t_8, 4, 1]$ and $R_2 = [n_1, n_8, t_2, t_9, 4, 1]$. We assume that each request requires a bandwidth equal to a wavelength. As there are no other reservations at this time, the links are reserved starting at the beginning

of the scheduling window of each request. A pictorial representation of the reservation table at t_0 is shown in figure 4

Node1	Node2	Wavelength	Start Slot	End Slot	Req ID
n ₁	n ₃	λ_1	t ₁	t ₄	R ₁
n ₁	n ₃	λ_2	t ₂	t ₅	R ₂
n ₃	n ₆	λ_1	t ₁	t ₄	R ₁
n ₃	n ₆	λ_2	t ₂	t ₅	R ₂
n ₆	n ₇	λ_1	t ₁	t ₄	R ₁
n ₆	n ₈	λ_1	t ₂	t ₅	R ₂

Table 1: Reservation Table at t_0

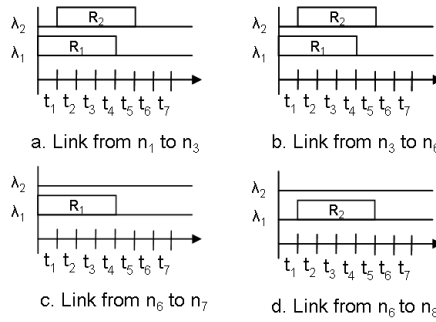


Figure 4: Reservation Table at t_0

Let us assume that a third request arrives at time t_1 for a path between n_1 and n_8 with $R_3 = [n_1, n_8, t_3, t_8, 3, 1]$. Since all the wavelengths along links $n_1 \rightarrow n_3$ and $n_3 \rightarrow n_6$ are busy till time t_4 , the shortest path $n_1 \rightarrow n_3 \rightarrow n_6 \rightarrow n_8$ is not available for slots t_3 and t_4 . But due to the large scheduling window, the request can be still accepted for slots t_5 , t_6 and t_7 for the same path.

At time t_2 a fourth request arrives with $R_4 = [n_1, n_7, t_3, t_5, 1, 1]$. In this case, all the wavelengths along links $n_1 \rightarrow n_3$ and $n_3 \rightarrow n_6$ are busy till t_5 and the shortest path $n_1 \rightarrow n_3 \rightarrow n_6 \rightarrow n_7$ is not available for all slots in the scheduling window. So in this case, another path has to be determined. This new path can be a 4-link path i.e. $n_1 \rightarrow n_2 \rightarrow n_5 \rightarrow n_6 \rightarrow n_7$ and the wavelengths that are reserved are λ_1 , λ_1 , λ_1 and λ_2 .

The state of the reservation table at time t_2 is given in figure 5

The objective in this paper is to determine a scheduling policy to route each incoming lightpath connection request dynamically while minimizing the probability that a connection request will be refused due to lack of available lightpaths and maximizing the overall network throughput.

3 Advance Network Scheduling and Management

In this section we discuss the various advance scheduling algorithms considered in this paper.

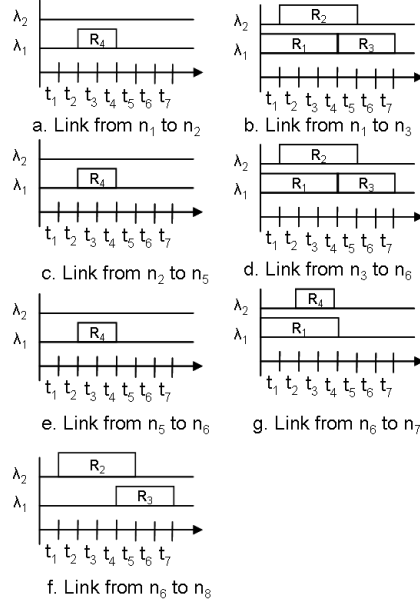


Figure 5: Reservation Table at t_2

3.1 Advance Reservation Scheduling Strategies

The Routing and Wavelength Assignment (RWA) [20, 21] problem deals with finding a route on the network topology and assigning a wavelength on each link of the selected route for every lightpath established over the network. Typically a connection request can be of three types [20] static, incremental, and dynamic.

With *static traffic*, the entire set of connections is known in advance, and the problem is then to set up lightpaths for these connections while minimizing network resources such as the number of wavelengths or the number of fibers in the network. The RWA problem for static traffic is known as the Static Lightpath Establishment (SLE) problem. The SLE problem can be formulated as a mixed-integer linear program [22], which is NP-complete [23]. To make the problem more tractable, the SLE problem can be partitioned into two subproblems, namely routing and wavelength assignment, and each subproblem can be solved separately. A review of these approaches is given in [20].

In case of *incremental traffic*, connection requests arrive sequentially and the lightpath established for each connection remains in the network indefinitely. Lastly for *dynamic traffic*, a lightpath is set up for each connection request as it arrives, and the lightpath is released after some finite amount of time. The objective in the incremental and dynamic traffic cases is to set up lightpaths and assign wavelengths in a manner that minimizes the blocking probability of a connection, or equivalently maximizes the number of connections that are established in the network at any time. This problem is referred to as the Dynamic Lightpath Establishment (DLE) problem. It is more complex to solve and usually heuristics are used to solve the routing and wavelength assignment subproblems separately [21].

There is another characterization of a traffic model where the setup and teardown times of the demands are *known in advance*. This is known as the Scheduled Traffic Model [24]. There can be a *fixed window* or a *flexible window* with this model. The start and end time of the connection cannot be altered in the fixed window model, but with the flexible approach the

start and end times can slide within a larger window. Integer Linear Program formulations and algorithms have been proposed to solve these problems in [18, 24].

In this paper, we consider *DLE requests* that belong to a *flexible scheduling window*. We have developed several strategies for searching the reservation schedule to determine whether a new request R can be accepted. These strategies use a combination of different scheduling decisions and search techniques. Our objective is to determine a scheduling algorithm that minimizes the blocking probability, i.e. the probability of not scheduling a request within its window, minimizes fragmentation in the usage of each wavelength and maximizes network utilization. We only consider *flexible advance reservations* as they have lower blocking probability than the fixed advance reservation model [19, 16].

We consider two categories of advance scheduling. In the first, a call is blocked if it cannot be scheduled within its requested flexible window. In the second, calls are not blocked, but rather delayed and scheduled at the first available time instance which maybe outside the requested window.

As will be seen below, we always determine the shortest path from a source to a destination using Dijkstra’s algorithm. Different costs associated with the links can be used. The most common link cost is the propagation delay. However, in order to balance the load in the network one can use a link cost that is determined based on the current and future usage. In this case we start with all the links having a cost of 1. As lightpaths are reserved, the link cost is incremented by the number of slots reserved on that link. The cost is decreased after the request is serviced. Hence the weights only reflect the current and future utilization of the link.

We propose the following scheduling strategies:

- **Switch Path First (SPF)**: In this scheme we start from the beginning of the scheduling window and check k different paths before sliding the window to the next time slot.
- **Slide Window First (SWF)**: In this scheme we check one path at a time for all of the scheduling window slots. If a path cannot be reserved during the scheduling window, the next shortest path is checked.

We also propose a variant of these strategies by allowing the weights of the links to vary based on current and future utilization of the link. So the weight of a link increases as it gets more and more advance reservations. Opposite, if the link does not have too many advance reservations then its cost will decrease. This provides load-balancing in the network. We call these variant scheduling strategies:

- **Load Balancing - Switch Path First (LB-SPF)**: In this scheme the weights of the links are based on current and future utilization of the link, but the search algorithm is the same as SPF.
- **Load Balancing - Slide Window First (LB-SWF)**: In this scheme the weights of the links are based on current and future utilization of the link, but the search algorithm is the same as SWF.

All the strategies can be used with both the wavelength assignment schemes *First-Fit* and *Min-Gap* described below in section 3.2. We now proceed to describe in detail the two algorithms SPF and SWF.

3.1.1 Switch Path First (SPF) Algorithm

In this scheme when a request R arrives, we first try to find the shortest available path starting at time s for d slots. This is done by first finding the shortest path, using a delay propagation

link cost. Then, we check if all the links on this path have a free wavelength for d slots starting at time $s + t$, where $t=0$. If any link is busy along the path, the topology is updated by removing that link and the next shortest path is determined. This step is repeated until either a path is available or a maximum of k different paths have been considered. If a path cannot be determined, we repeat the whole process with a start time equal to $s + t$, where $t=1$. t is incremented by one slot each time until an available path is found or $t = e - s - d$, whereupon a request is blocked.

Algorithm 1 The SPF algorithm

```

1: function FINDPATH(REQUEST R, TOPOLOGY T)
2:  $i = 1$ 
3: start time =  $s$ 
4: end time =  $s + d$ 
5:   while (end time  $\leq e$ ) do
6:     while ( $i \leq k$ ) do
7:       find shortest path with Dijkstra's algorithm with propagation delay link cost
8:       if a path is found then
9:         if wavelengths are available on all links during start time and end time then
10:          assign wavelengths, update all tables
11:          return
12:         else
13:          delete busy link from the topology
14:           $i ++$ 
15:         end if
16:       end if
17:     end while
18:   start time = start time +  $t'$ 
19:   end time = start time +  $d$ 
20: end while
21: end function

```

3.1.2 Slide Window First (SWF) Algorithm

In this algorithm, we try to find a free period d starting at $s + t$, where $t = 0, 1, 2, \dots, e - s - d$. If the first shortest path is not free for the required duration during the window, the busiest link defined as the one that uses the maximum number of slots during the scheduling window, is removed from the network topology and the procedure is repeated until either an available path is found or a maximum of k paths is considered.

3.2 Wavelength Assignment

It is possible that multiple wavelengths on a particular link along the source-destination path, are available for the advance reservation. In this case, the scheduling algorithm must determine which wavelength to allocate from the pool of available wavelengths. We use the following two wavelength allocation schemes:

First-Fit : This is a well known method for wavelength assignment as it gives good performance with least computational time. In this scheme, all wavelengths are arbitrarily numbered. When searching for available wavelengths, we always start from wavelength 1 and proceed sequentially to the last wavelength until a free wavelength is found. By searching the wavelengths

Algorithm 2 The SWF algorithm

```
1: function FINDPATH(REQUEST R, TOPOLOGY T)
2:  $i = 1$ 
3:   while ( $i \leq k$ ) do
4:     start time =  $s$ 
5:     end time =  $s + d$ 
6:     find shortest path with Dijkstra's algorithm with propagation delay link cost
7:     if A path is found then
8:       while (end time  $\leq e$ ) do
9:         if wavelengths are available on all links during start time and end time then
10:          assign wavelengths, update all tables
11:          return
12:        else
13:          start time = start time +  $t'$ 
14:          end time = start time +  $d$ 
15:        end if
16:      end while
17:    else
18:      remove the busiest link during the window from topology
19:    end if
20:     $i ++$ 
21:  end while
22: end function
```

in this manner, connections are packed into a smaller number of wavelengths thus freeing other wavelengths for reservations. First-fit does not require global knowledge of the network and no storage is needed to keep the network state. This scheme performs well in terms of blocking probability and fairness, and is preferred in practice because of its small computational overhead and low complexity [20].

Min-Gap : This scheme aims at reducing the fragmentation in wavelength usage. With advance reservations we have the information of the wavelength usage with the associated times, and hence we can use this information when assigning the wavelength in order to reduce the gaps between reservations. The main drawback of this scheme is the additional computational overhead when searching for the least gap throughout the existing reservations.

There can be three different ways to minimize the gaps:

- **Min-Leading-Gap**: This wavelength assignment scheme minimizes the unused leading gaps on a wavelength.
- **Min-Trailing-Gap**: This scheme minimizes the unused trailing gaps on a wavelength.
- **Best-Fit**: This scheme minimizes the sum of the leading and trailing gaps.

An example of these wavelength allocation schemes is shown in figure 6. We can see that the new lightpath (LP) reservation can be accommodated by all the four wavelengths. *First-Fit* will select λ_1 . λ_2 minimizes the leading gap between the new and previous reservations therefore it will be selected by *Min-Leading-Gap* scheme, while λ_3 minimizes the trailing gaps so it will be selected by *Min-Trailing-Gap*. The *Best-Fit* will select λ_4 , as it minimizes the sum of leading and trailing gaps. A comparison of these three schemes is presented in section 4.

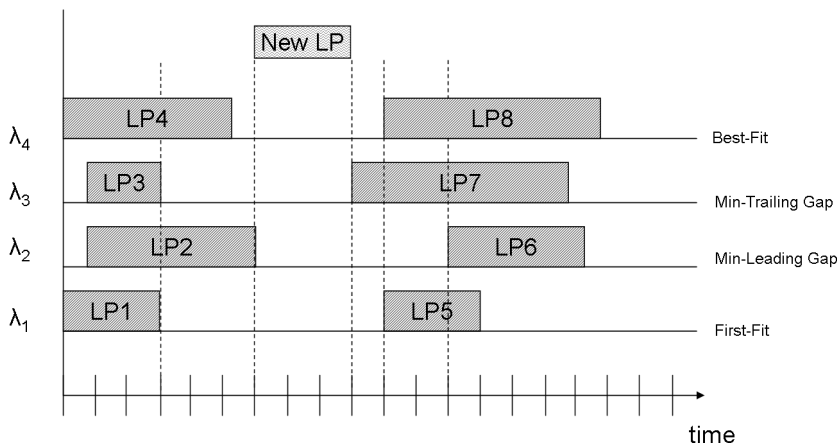


Figure 6: Wavelength Assignment using Min-Gap

3.3 Network Failures

Accepted advance reservations can be affected by network failures. In view of this, a strategy is required in order to restore the existing network connections and re-schedule future reservations.

We consider network component failures, which could be either link failures or node failures. When a link fails, all its constituent wavelengths will fail. Link failure is more of a concern because most nodes have built in redundancy. Because of the high data rate on these networks, it is necessary to develop appropriate path protection and restoration schemes to prevent or reduce data loss. In case of protection, a path or a link is protected against failures by pre-signing resources for a backup path, while in restoration schemes an alternate route is discovered dynamically for each failed connection after the failure occurs. Protection schemes have faster recovery time and provide guaranteed recovery but they require more network resources. Therefore due to lower costs involved, we use the restoration mechanism. Given a failure, the question of interest is how far into the future should we reschedule the existing reservations.

In order to deal with network failures the DNRM must be aware of the current network state. This information can be obtained from the network devices in the form of a link state database using SNMP or other management protocols. Routing protocols, such as OSPF react slowly to network failures. In view of this, monitoring tools that talk directly to the network devices to get the latest state of interfaces and ports should be used as well.

As soon as a link failure is observed, all the in-service connections using this link have to be re-scheduled. Also, it is useful to reroute the requests that are scheduled but not yet in service that use the failed link. These are likely to be affected if the failed link is not restored on time. The straight forward approach is to re-schedule all the future connections that use the failed link. However this may not be good solution because if the link is repaired early, the connections will be using sub-optimal paths. Thus, a re-routing interval has to be determined to re-schedule future reservations so that to minimize the number of dropped reservations. We note that the reservations can be dropped when re-scheduling them, if a free path cannot be found for the start time that a reservation has already been made.

Since the failure duration is not known in advance, the re-routing interval has to be estimated using different methods. From [25] it is clear that substituting the exact knowledge of the down-

time by vague estimations is dangerous, since this leads to a significantly worse performance. Also in [25] two techniques were presented that are independent of the actual downtime. These are load-based and feedback based. In the case of load-based approach the current load and the booking profile during the next time slot should be less than a threshold value η which depends on the network topology. While in the feedback based method, the initial re-routing interval is one slot and it is increased based on the percentage of successfully re-routed reservations during the last slot. The interval duration is increased until the percentage of successfully re-routed reservations is sufficiently high.

In this paper, we have used an approach in which the re-routing interval is based on the knowledge of the duration of previous failures. Specifically, we maintain a *moving average of the historical failure durations* and use this as an estimate of the restoration time. Similar to the feedback based approach, if the link does not come up at the end of the current re-routing interval, we increase the re-routing interval by an amount equal to the previous re-routing interval.

3.4 Network Optimization

An optimization of the network can be performed by re-scheduling the lightpath requests. This can be done in two ways: either by changing paths only or by changing both the path and the start time. The latter is not very desirable as it involves re-negotiating the time with the users, and in view of this we will not be considering it in this study.

The optimization techniques have been discussed in several papers. Most of the authors have proposed solutions for static traffic demand and heuristics for long term on-demand traffic flows [26, 27]. In [28] the authors have proposed an optimization scheme for advance reservations by re-configuring them without changing their reservation times. Using this scheme, if a request is blocked, all the existing connections that time-overlap with this request are unscheduled and then re-scheduled after sorting them according to their start times. This may result in provisioning of all the requests including the one which was blocked earlier. If this does not work, all connections are restored to their original state, i.e., no re-configuration takes place.

We use the same technique but instead of doing the re-configuration after every blocked request, we do it periodically by re-scheduling all the requests for a specific number of slots in future.

4 Performance Evaluation

4.1 The Simulation Model

We have conducted our simulation experiments on a 14 node NSF-NET topology with 42 uni-directional links and a 33 node GEANT network topology with 94 uni-directional links as shown in figure 7. We assume 10 wavelengths on each link and full wavelength conversion at each node. The time is slotted with the duration of each slot being 30 minutes.

We assume that requests arrive in a Poisson fashion and all requests need to reserve a lightpath with bandwidth equal to one wavelength. The duration of a reservation is uniformly distributed. The start times of the request are generated within a window of 400 slots. To simulate a more realistic environment, we have generated the intermediate period between the arrival of the request and the start of the reservation using the discrete probability distribution shown in figure 8. We assume that more requests will be for the reservation slots in the near future i.e next 24 slots and very few requests will be for reservation slots far into the future

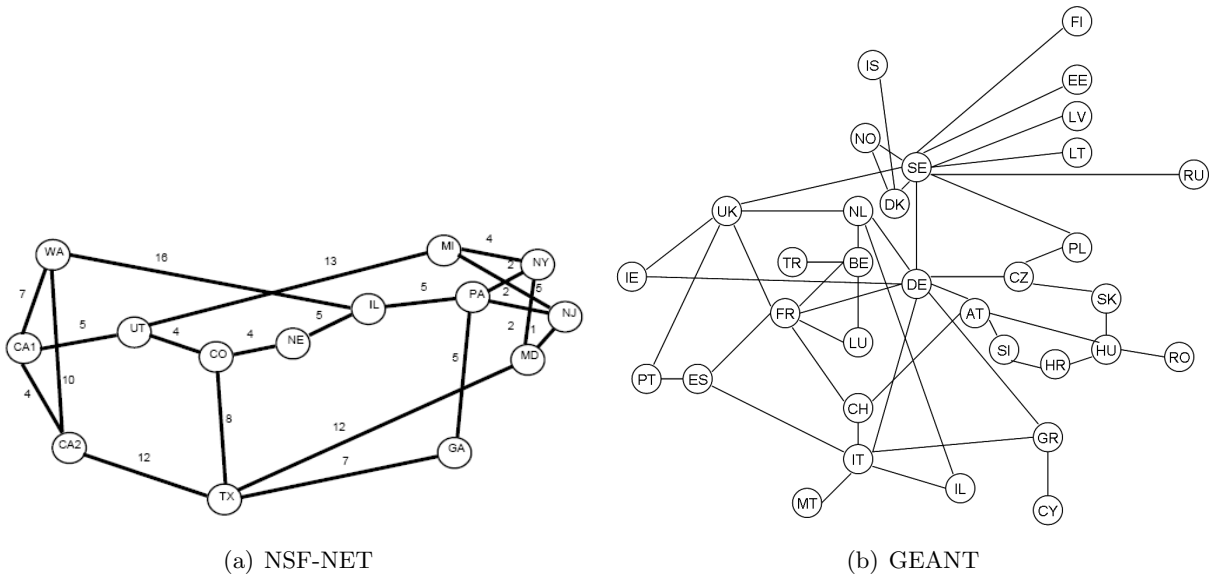


Figure 7: Network Topologies

e.g. after two or three days. The source and destination nodes for the requested connection are selected randomly using a uniform distribution.

We assume that the scheduling window is twice the reservation duration, i.e., $(e - s) = 2d$, based on the results in [19] where the authors have shown that using just 1 or 2 units of flexibility improves the performance significantly. Our results agree with this observation. Figure 9 gives the blocking probability for different window sizes. The solid curve marked as $1d$ corresponds to the case $e - s = d$. The dotted line curves correspond to the cases when $e - s = 2d$ and $e - s = 3d$. We note that the scheduling window with a width of twice the duration has much lower blocking probability than without any flexibility. Also increasing it further does not improve the performance to a large extent but the delay between the start time of the actual reservation and the start of the window increases. We ran the simulations under different network loads, where network load is determined by the request arrival rate and the reservation durations. The parameter of interest is the blocking probability B_p .

We have used a failure model with link failures only. In our simulation, we randomly select any link in the network as a failed link. The mean time to failure is exponentially distributed with a mean of 80 slots. The recovery time is also exponentially distributed with a mean of 48 slots. We assume that when a link fails all wavelengths on that link fail.

The simulations were run for long time durations with large number of arrivals such that a sufficiently small confidence interval within 1% of the mean with 95% confidence is reached. The confidence intervals are not given in the graphs as they are not discernible. Finally, the simulation model was written in Java.

4.2 Numerical Results

4.2.1 Scheduling Algorithms

In this section we will present the simulation results and comparison of our scheduling algorithms for advance reservation of lightpaths.

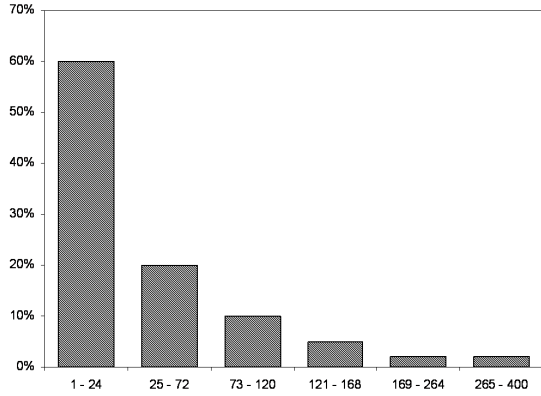


Figure 8: Discrete Probability Distribution for the Intermediate Time between the Request Arrival Time and Reservation Time

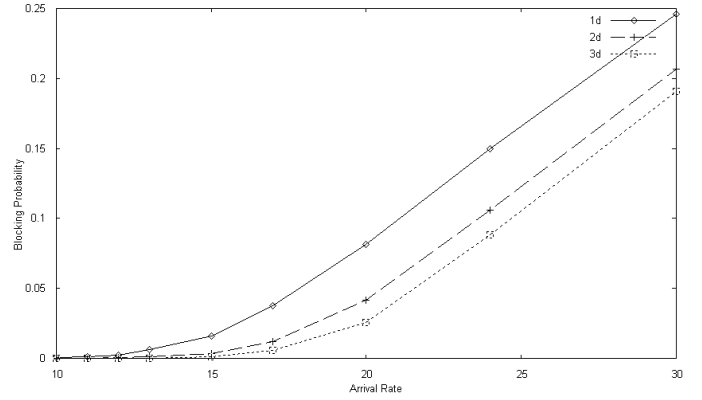
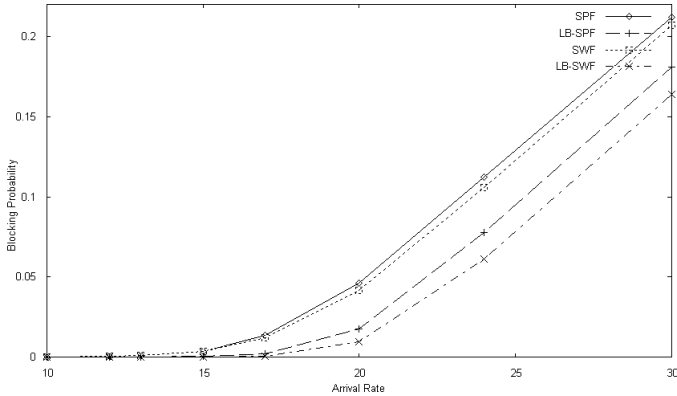
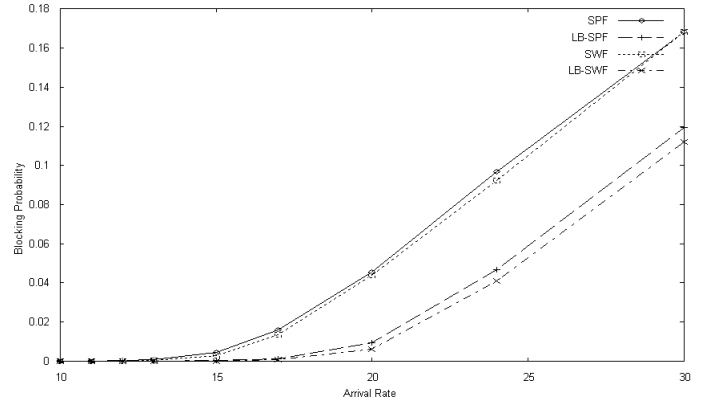


Figure 9: Flexibility in Scheduling Window



(a) NSF-NET



(b) GEANT

Figure 10: Arrival Rate vs. Blocking Probability

Figures 10a and 10b show the effect of the arrival rate on the blocking probability. The rate is expressed in terms of number of requests/slot. From the graphs, we can see that SWF performs slightly better than SPF because it tends to schedule the connections on shorter paths. We also observe a significant drop in the blocking probability for both SWF and SPF schemes when load-balancing is used. The link utilization comparison between SWF and LB-SWF for NSF-NET is shown in figure 11. These results are for an arrival rate of 60 requests/slot. The slope of the curves show how the load is balanced among the links with and without load balancing. The links are sorted in the order of utilization. We also note that for most of the links, LB-SWF achieves lower utilization than SWF. This is because the load is balanced among the links.

In our algorithms we try k alternate paths before blocking a request. To find the optimum value of k , we ran the simulation with different values of k . The results are shown in figures 12a and 12b. It was observed that the blocking probability decreases initially but then it slightly increases as k increases. As we increase the value of k , longer paths are reserved which indirectly affects the overall blocking probability. When $k=1$, SPF has same blocking probability as SWF because there are no alternative paths. The value of k with the least blocking probability varies

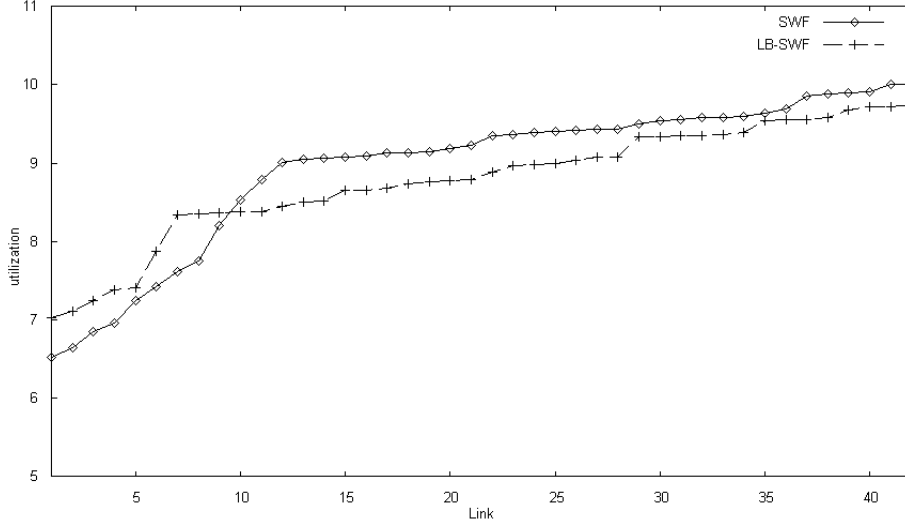


Figure 11: Link Utilization : SWF vs LB-SWF

with the network topology. It depends on the number of links in the network and is independent of the link costs. We have used the optimum value of k for each scheme for the rest of the experiments e.g. it is 3 for SPF and 4 for SWF for NSF-NET but for GEANT it is 4 for SPF and 5 for SWF.

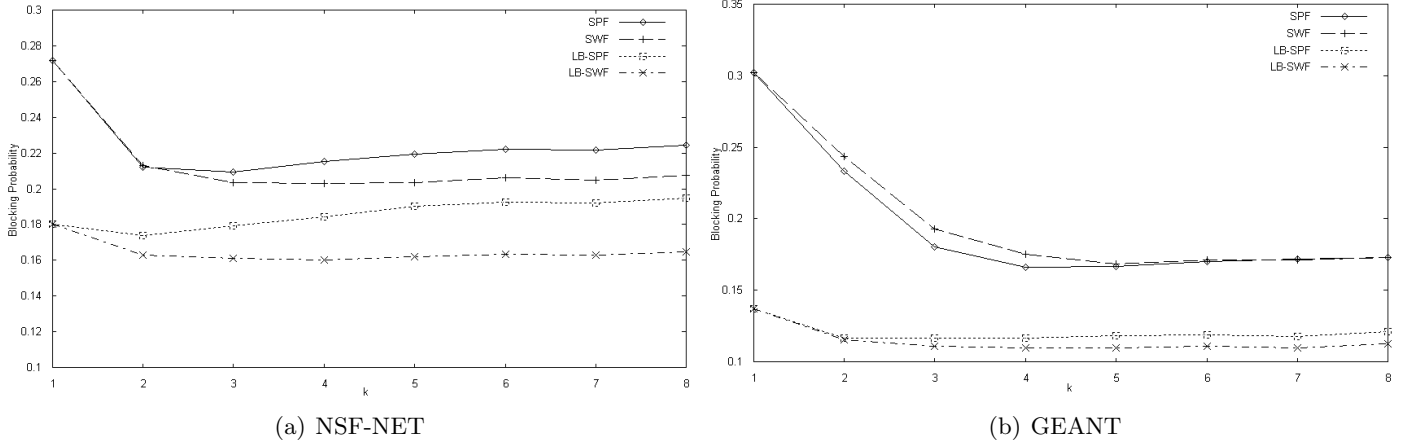


Figure 12: k vs Blocking Probability

The graphs in figures 13a and 13b show the effect of the connection duration d on the blocking probability. The duration is uniformly distributed with a minimum of one slot and a maximum d_{max} . We can see that the blocking probability increases as d_{max} increases. SWF with load balancing still performs the best. We also observe that the blocking probability significantly increases for $d_{max} > 5$ for the considered topologies.

Figures 14a and 14b show the reservation delay, i.e., the time elapsed from the requested start time s to the time $s + t$ where the reservation was actually made, as a function of d_{max} for both SPF and SWF. We see that SPF always tries to schedule as close to the start time s of the scheduling window as possible.

We have also implemented a non-blocking version of the scheduler. In this case the requests

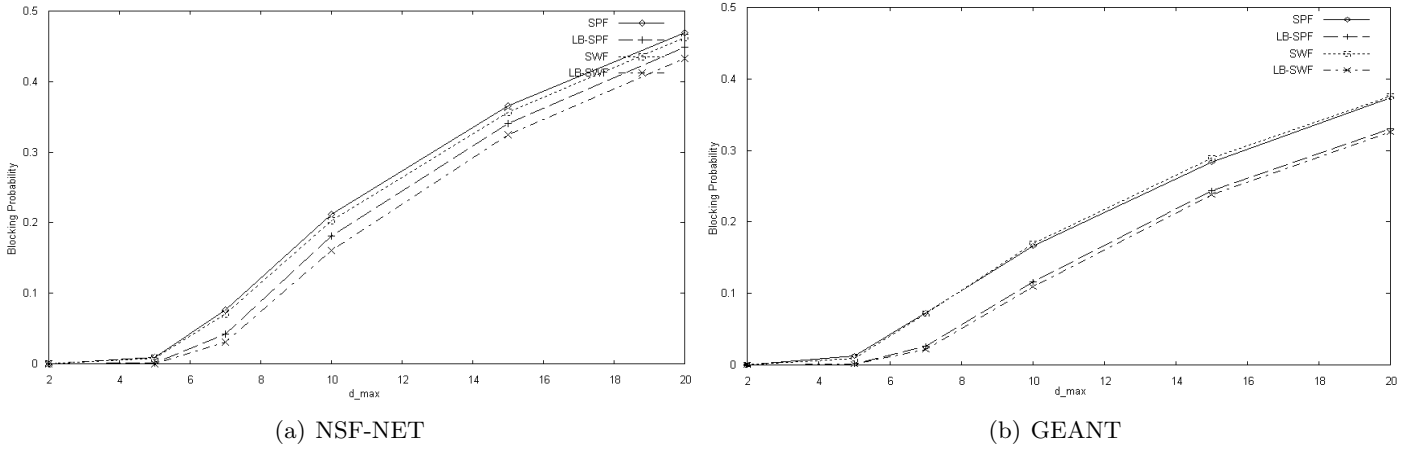


Figure 13: d_{max} vs Blocking Probability

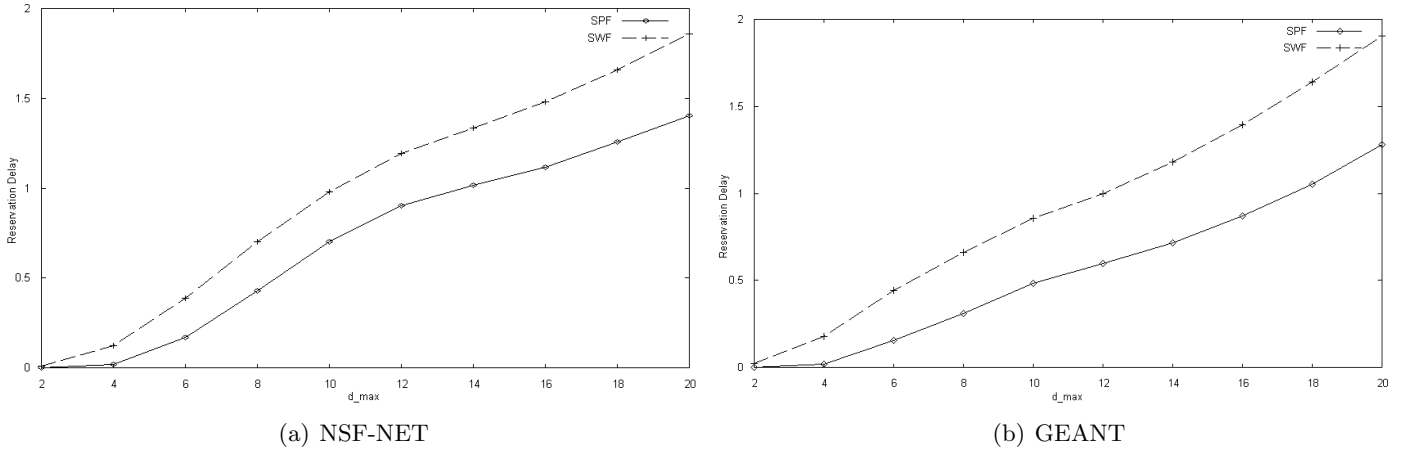


Figure 14: Reservation Delay vs d_{max}

are never blocked and scheduled at the first available time which can be outside the scheduling window. In this case, both the SPF and SWF algorithms keep on sliding the window until they find an available path. The difference between the two approaches is that SPF checks k different paths before sliding the window by one slot, while SWF slides the window by $2d$ slots before checking another path. The results are given for NSF-NET in figure 15. In figure 15a, the average reservation delay is plotted versus the arrival rate for all four scheduling schemes. In figure 15b, the percentage of reservations outside the scheduling window are plotted versus the arrival rate for all the four schemes. We can see that at high arrival rates the network becomes unstable and around 90% [figure 15b] of the requests are reserved at a time outside the scheduling window. Also the delay increases tremendously to an average of 500 slots which is not desirable [figure 15a]. But at lower rates there is a low percentage of connections scheduled outside the window and also with a low reservation delay. We can see the SPF and LB-SPF schemes give better results because they tend to reserve the request at a time close to the start of the scheduling window. Obviously in case of a non-blocking scheduler, a stability condition needs to be developed in order to protect the network from becoming unstable, i.e., the reservation delay becomes very large. Such a condition is not necessary for a blocking scheduler, where a

request will get blocked if it cannot be scheduled within its requested window $[s,e]$.

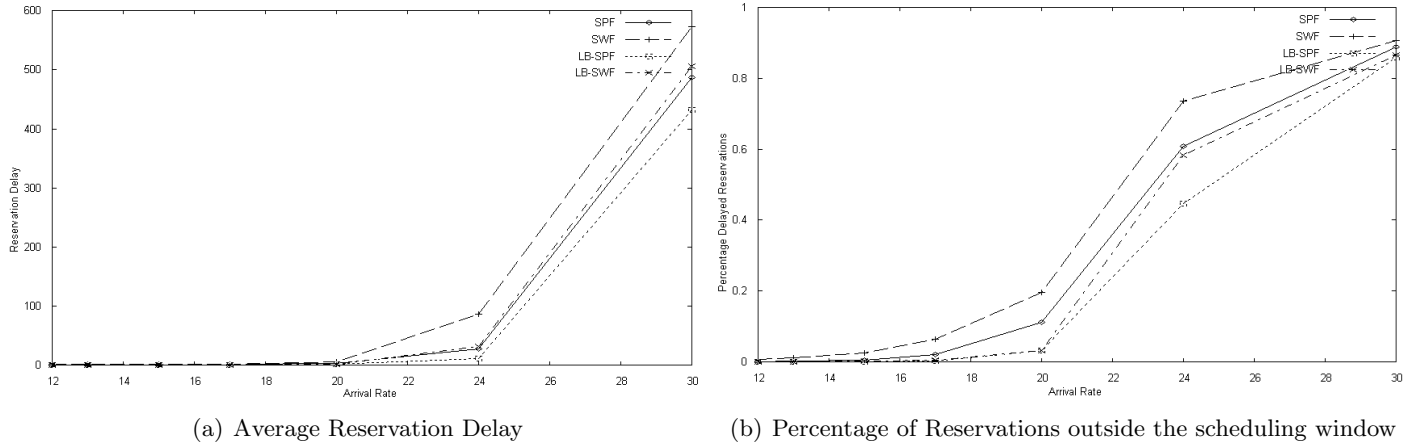


Figure 15: Non Blocking Scheduler

For wavelength assignment, we have implemented two approaches First-Fit and Min-Gap as discussed earlier. We can see [figure 16] that the Min-Gap scheme with minimum leading and trailing gap give the lowest blocking. The First Fit scheme also performs well. The Best-Fit Min-Gap scheme has the worst performance because it leaves small un-used gaps before and after a reservation. These small gaps cannot be used and this results in more fragmentation.

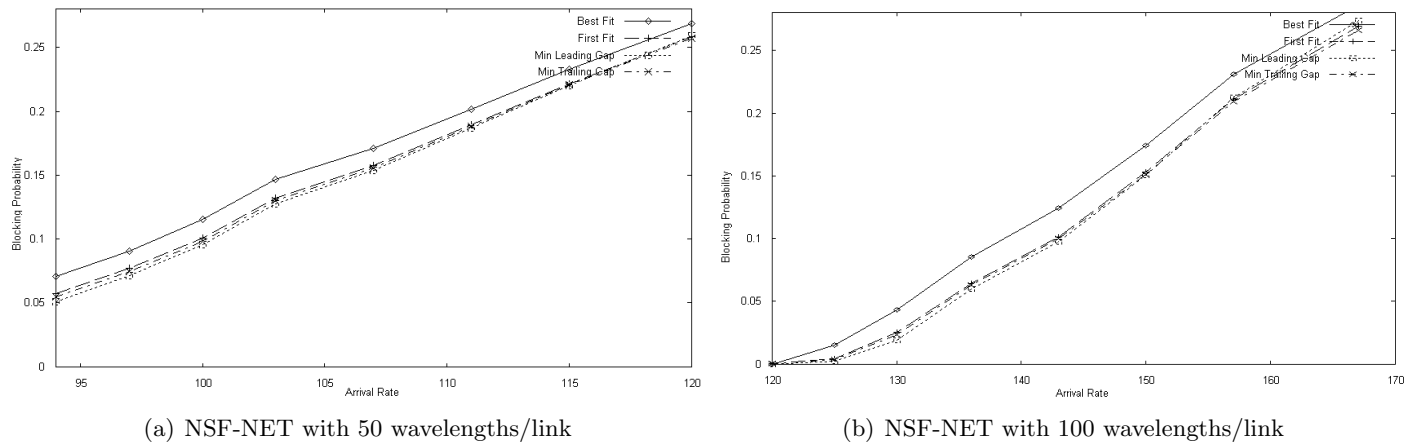


Figure 16: Wavelength Assignment

We have studied how the requests with long and short durations affect each other. For this we generated requests which have a maximum duration $d_{max}=2$ with 50% probability and $d_{max}=20$ with 50% probability. We compared the results of blocking probability of both classes of requests with the results when they are not in the presence of each other. We ran the simulation experiment for NSF-NET. The requests arrive in a Poisson fashion. The advance reservation scheme used is LB-SWF with First-Fit. In figure 17, the blocking probability curves are plotted against the arrival rate for a) all requests have $d_{max}=20$. b) all requests have $d_{max}=2$. c) 50% of the requests have $d_{max}=2$ and 50% of the requests have $d_{max}=20$. We have two curves for the last case, one for each class of request. These are labeled as "Mixed $d_{max}(2)$ " and "Mixed $d_{max}(20)$ ". We see that both classes of requests affect each other and the blocking probability

increases for both. The blocking probability of requests with short duration, i.e., $d_{max}=2$ only was zero before and it increases with the presence of requests with $d_{max}=20$. Same is the case with the long requests.

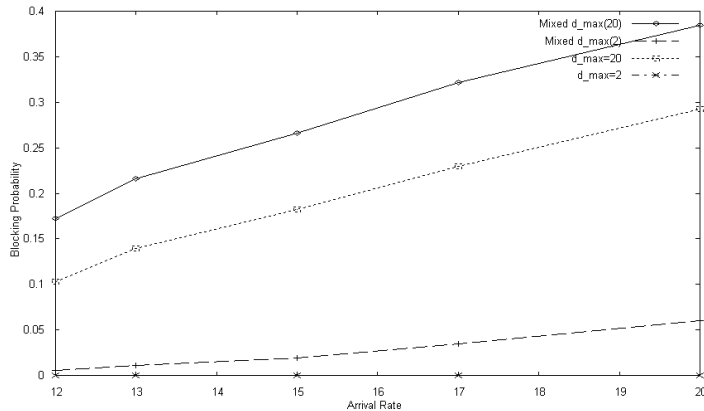


Figure 17: Requests for long and short durations simultaneously

4.2.2 Failure Recovery

We used a failure model with link failures only. We assume that when a link goes down all wavelengths on that link fail. Only one link can fail at a time and this link is randomly selected from all the links. The mean time to failure is exponentially distributed with a mean of 80 slots. The recovery time is also exponentially distributed with a mean of 48 slots.

The parameters of interest are the percentage termination and the overhead. Percentage termination is the percentage of reserved connections that cannot be re-routed and are dropped. The overhead is defined as the percentage of reserved connections that are unnecessarily re-routed as they were un-affected by the failure because the reservation start time turns out to be after the instant that link is repaired. Also we are interested in how the overall blocking probability is affected by the failure recovery process.

We calculate the re-routing interval using the following three policies

1. The **Fixed** re-routing interval with feedback: The length of the re-routing is fixed. When the link goes down the connections are re-routed for this fixed interval. If the link does not come up at the end of the interval, the re-routing interval is increased by an amount equal to the fixed value.
2. **Adaptive** re-routing interval with feedback: In this case the re-routing interval is calculated using a moving average of the historical recovery times. If the link does not come up at the end of the interval, the re-routing interval is increased by an amount equal to the last re-routing interval length.
3. **All** future reservations are re-routed: In this case there is no re-routing interval, all the reservations on the failed link are re-routed.

We compare all the three approaches with the ideal scenario where the failure duration is known and only the flows affected are re-routed.

In figure 18a the results are shown for NSF-NET. We can see that the longer the re-routing interval the lower is the termination percentage, because at an earlier time more alternative paths

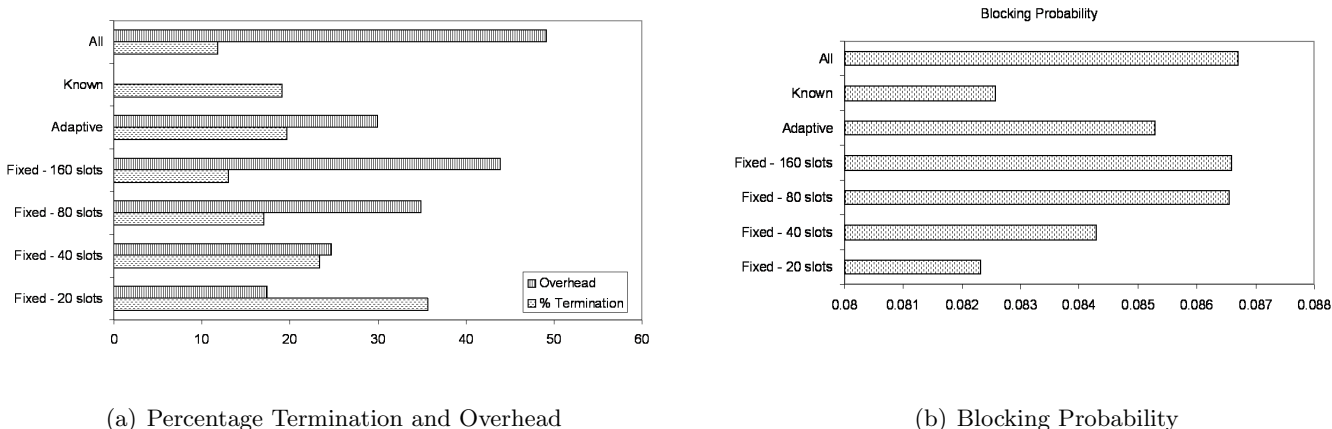


Figure 18: Failure Recovery

are free and the reservations can be re-scheduled. If the reservations are re-routed just before they are to start, the remaining alternate routes are more likely to be busy. But the overhead increases with the longer re-routing intervals. Since both the requirements are conflicting, there is a tradeoff between the two and best results are achieved when we use the adaptive method for determining the re-routing intervals. Having very long re-routing intervals also results in an increase in the overall blocking probability in the network due to a large number of sub-optimal paths [see figure 18b].

4.2.3 Periodic Re-configuration

In order to improve the network performance, we have implemented periodic re-configuration of existing reservations. The network topology considered is NSF-NET. The re-configuration process occurs every 24 slots and re-configures the existing reservations for the next 24 slots without changing the reservation times. This is done by sorting the requests in an ascending order of their start times and then re-scheduling them one at a time starting from the request with the earliest start time. This process can result in terminations, in which case we restore the reservation table to its last state (i.e. no re-configuration). We observed that the re-configuration process failed most of the time, i.e., the reservation table was restored to its last state due to at least one termination, especially at high rates. We also observed that if we allowed for a small percentage of terminations, the process succeeded most of the time.

Figure 19a shows the blocking probability as a function of the arrival rate with and without re-configuration. We can see that the overall blocking probability is slightly improved with the re-configuration. Also re-configuration which allows 2% or 5% blocking of the reserved connections, lying in the re-configuration period, further improved the overall blocking probability in the network. The overall blocking probability also includes the terminations during the re-configuration. In figure 19b, the percentage of time the re-configuration process fails is plotted against the arrival rate. The results show that at higher arrival rates the re-configuration process fails more frequently, but this improves significantly if some amount of termination is permitted. With 5% allowed termination the re-configuration succeeds all the time even at higher arrival rates.

From the results, we can conclude that using this method of periodic re-configuration for

offline optimization has very little impact. This may not be the case, should a more efficient re-configuration algorithm be developed.

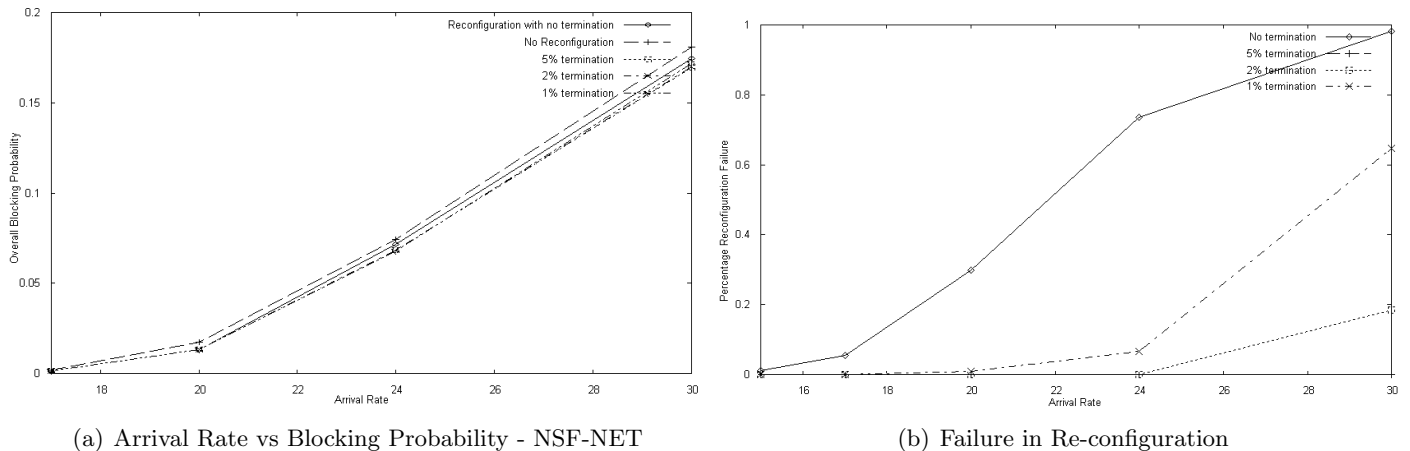


Figure 19: Periodic Reconfiguration

5 Conclusion

In this paper, we have evaluated various algorithms for advance scheduling of lightpaths that can be implemented in a Grid resource scheduler. The scheduling of lightpaths involve both routing and wavelength assignment. Our simulation results show that minimum cost adaptive routing where link costs are determined by the current and future usage of the link provides the minimum blocking. Moreover, searching for k alternate paths within the scheduling window significantly improves the performance. For wavelength assignment, a scheme that minimizes unused leading or trailing gaps gave the best results.

We have also analyzed several approaches for failure recovery. In all these approaches a re-routing interval is determined and all the reservations on the failed link during this period are re-scheduled. The results indicate that a short re-routing interval gives large number of terminated connections while a long interval re-routes unaffected connections. There is a tradeoff between the two and the best results are obtained when the re-routing interval is based on the moving average of the historical failure times and is updated continuously.

The network utilization can be improved further by offline optimization of the reserved connections that are not in service yet. In our simulations, we have implemented periodic re-configuration of reserved lightpaths. This improves the performance but is not very significant. More work needs to be done in this area for further improvement of performance.

6 Acknowledgements

We acknowledge Yufeng Xin, John Moore from MCNC and Jon McLaren from LSU with the concepts and discussions. We would also like to thank Carla Hunt from MCNC for her help with the use of resources for the simulation experiments.

References

- [1] DynaCode web page. <http://www.nsf.gov/awardsearch/showAward.do?AwardNumber=0540374>.
- [2] Andrei Hutanu, Gabrielle Allen, Stephen D. Beck, Petr Holub, Hartmut Kaiser, Archit Kulshrestha, Miloš Liška, Jon MacLaren, Luděk Matyska, Ravi Paruchuri, Steffen Prohaska, Ed Seidel, Brygg Ullmer, and Shalini Venkataraman. Distributed and collaborative visualization of large data sets using high-speed networks. *Future Generation Computer Systems: The International Journal of Grid Computing: Theory, Methods and Applications*, 8:1004–1010, Oct. 2006.
- [3] Alvaro Saurin Ladan Gharai, Tom Lehman and Colin Perkins. Experiences with high definition interactive video conferencing. In *Proceedings of IEEE International Conference on Multimedia and Expo (ICME)*, Toronto, Canada, July 2007.
- [4] Enlightened computing project website. <http://www.enlightenedcomputing.org>.
- [5] G-lambda project website. <http://www.g-Lambda.net>.
- [6] Phosphorus project website. <http://www.ist-phosphorus.eu>.
- [7] Global lambda integrated facility (glif) website. <http://www.glif.is>.
- [8] Dragon web page. <http://dragon.east.isi.edu/twiki/bin/view/Main/WebHome>.
- [9] Cheetah web page. <http://cheetah.cs.virginia.edu/>.
- [10] User-controlled lightpath website. <http://www.uclp.ca>.
- [11] Maui scheduler website. <http://www.clusterresources.com/pages/products/maui-cluster-scheduler.php>.
- [12] Jun Zheng and Hussein T. Mouftah. Routing and wavelength assignment for advance reservation in wavelength-routed wdm optical networks. In *IEEE International Conference on Communications, ICC 2002*, volume 5, pages 2722– 2726, 2002.
- [13] Lars-Olof Burchard. Networks with advance reservations: Applications, architecture, and performance. *Journal of Network and Systems Management*, 13(4), December 2005.
- [14] I. Foster, C.Kesselman, C. Lee, R. Lindell, K. Nahrstedt, and A. Roy. A distributed resource management architecture that supports advance reservations and co-allocation. *7th International Workshop on Quality of Service (IWQoS)*, pages 27 – 36, 1999.
- [15] Chiara Curti, Tiziana Ferrari, Leon Gommans, S. van Oudenaarde, Elisabetta Ronchieri, Francesco Giacomini, and Cristina Vistoli. On advance reservation of heterogeneous network paths. *Future generation Computer Systems*, 21(4):525 – 538, April 2005.
- [16] Lars-Olof Burchard, Hans-Ulrich heiss, and Ceaser A. F. De Rose. Performance issues of bandwidth reservation for grid computing. In *Proceedings of 15th Symposium on Computer Architecture and High Performance Computing*, 2003.
- [17] Bin Wang, Tianjian li, Xubin Luo, Yuqi Fan, and Chunsheng Xin. On service provisioning under a scheduled traffic model in reconfigurable wdm optical networks. *2nd International Conference on Broadband Networks*, 1:13 – 22, October 2005.

- [18] A. Jaekel. Lightpath scheduling and allocation under a flexible scheduled traffic model. *GLOBECOMM 2006*, 2006.
- [19] Eric He, Xi Wang, and Jason Leigh. A flexible advance reservation model for multi-domain wdm optical networks. *IEEE GRIDNETS 2006*, October 2006.
- [20] H. Zang, J.P. Jue, and B. Mukherjee. A review of routing and wavelength assignment approaches for wavelength-routed optical wdm networks. *Optical Networks Magazine*, 1:47–60, January 2000.
- [21] H. Zang, J. Jue, L. Sahasrabudde, R. Ramamurthy, and B. Mukherjee. Dynamic lightpath establishment in wavelength routed networks, 2001. *IEEE Communications Magazine*, 39(9):100–108.
- [22] Rajiv Ramaswami and Kumar N. Sivarajan. Routing and wavelength assignment in all-optical networks. *IEEE/ACM Transactions on Networking*, 3(5):489–500, 1995.
- [23] I. Chlamtac, A. Ganz, and G. Karmi. An approach to high-bandwidth optical wans. *IEEE Transactions on Communications*, 40(7):1171–1182, July 1992.
- [24] J. Kuri, N. Puech, M. Gagnaire, E. Dotaro, and R. Douville. Routing and wavelength assignment of scheduled lightpath demands. *IEEE JSAC*, 21(8):1231–1240, October 2003.
- [25] Lars-Olof Burchard, Barry Linnert, and Joerg Schneider. Rerouting strategies for networks with advance reservations. *e-Science 2005*, pages 446–453, 2005.
- [26] Eric Bouillet, Jean-Francois Labourdette, Ramu Ramamurthy, and Sid Chaudhuru. Light-path re-optimization in mesh optical networks. pages 437–447, 2005.
- [27] Randeep Bhatia, Murali S. Kodialam, and T. V. Lakshman. Fast network re-optimization schemes for mpls and optical networks. *Computer Networks* 50(3), pages 317–331, 2006.
- [28] Xi Yang, Lu Shen, Ajay Todimala, B. Ramamurthy, and T. Lehman. An efficient scheduling scheme for on-demand lightpath reservations in reconfigurable wdm optical networks. In *Optical Fiber Communication Conference 2006 and the 2006 National Fiber Optic Engineers Conference*, page 32, 2006.