

Efficient Distributed Algorithm for RWA Using Path Protection

A. Jaekel and Y. Chen

University of Windsor/Computer Science, Windsor, Canada

Email: arunita@uwindsor.ca

Abstract—A number of Integer Linear Program (ILP) formulations for both static and dynamic lightpath allocation have been proposed, for the design of survivable WDM networks. However, such formulations become computationally intractable for larger networks. In this paper, we present two heuristic algorithms for dynamic lightpath allocation, using both dedicated and shared path protection. The first is a centralized algorithm and the second is a distributed algorithm. The objective in both cases is to minimize the amount of resources (wavelength-links) needed to accommodate the new connection. We have tested our algorithms on a number of well-known networks and compared their performance to “optimal” solutions generated by ILPs. Experimental results show that our heuristics generate solutions that are within 15% of the optimal. Our approach is much faster and more scalable compared to existing ILP formulations.

Index Terms—Distributed RWA; Path protection; Survivable WDM networks

I. INTRODUCTION

The design of robust, and reliable optical networks [1] has been a topic of intense research interest in recent years. Wavelength division multiplexing (WDM) divides the tremendous bandwidth of a fiber into K non-overlapping carrier wavelengths, called WDM channels. A lightpath in an optical network is an end-to-end all-optical communication path from a source node to a destination node through a number of intermediate router nodes. Each lightpath must be assigned a route over the physical network, and a specific channel on each fiber it traverses. This is the standard routing and wavelength assignment (RWA) problem. A wavelength routed optical network may use either a static or a dynamic lightpath allocation strategy. In static allocation, the set of lightpaths to be established is known in advance [2]. If there are no wavelength converters at the intermediate nodes, a lightpath must be assigned the same channel on each fiber. This is called the wavelength continuity constraint, and is followed in this paper.

A number of ILP formulations for solving the static and dynamic RWA formulations have been presented. However, these formulations are usually very complex and can give optimal solutions, only for smaller networks. They provide interesting insights into the design problem, but are not feasible for practical

networks. In this paper, we first present a heuristic algorithm which assumes there is a node which acts as a “central agent” and has complete knowledge of the entire network. This follows similar assumptions used by existing ILP formulations. The main problem with such a centralized algorithm is that the central agent can quickly become a bottleneck. We then present a distributed algorithm that allocates resources based only on local knowledge available at each node. In both cases, we use path protection techniques. This means that for each new connection, a primary path and an edge-disjoint backup path are established during call setup. We demonstrate through simulations that our algorithms generate solutions comparable to the optimal solutions (generated by the ILP formulations) but are much faster and more scalable than exact ILP formulations.

The remainder of this paper is organized as follows. In section 2, we briefly review existing formulations for static and dynamic lightpath allocation and path protection techniques for WDM networks. In sections 3 and 4 we present our centralized and distributed algorithms for dynamic lightpath allocation. In section 5 we present and analyze our experimental results and conclude with a critical summary in section 6.

II. REVIEW

In dynamic allocation, a lightpath is created in response to a request for communication from a source s to a destination d by determining a path in the network and then allocating a usable WDM channel for every fiber in the path [3]. When the communication is over, the WDM channels used for this communication are reclaimed for future use in some other communication. A dynamic scheme does not guarantee that communication from a source to a destination will always be possible. The routing and wavelength assignment (RWA) problem is a difficult problem because of the inherent limitation on the number of wavelengths that can be used on a single fiber and the fact that every lightpath sharing the same fiber must be assigned a different wavelength. When a new connection request arrives, it is important to try to accommodate the request in a way that optimizes the use of available resources. One common design objective is to minimize the number of “wavelength-links”. One wavelength-link corresponds to the assignment of one WDM channel on a single fiber. So, if

a lightpath is assigned a channel w_k (following the wavelength continuity constraint), and it traverses three physical fibers from its source s to its destination d , then it uses three wavelength-links. In [4] authors provide an excellent review of the routing and wavelength assignment (RWA) problem for various different physical topologies. An ILP formulation for dynamic lightpath allocation is presented in [5]. A centralized heuristic to solve the RWA problem is given in [6]. Optimal RWA for lightpath allocation under a differentiated service model is discussed in [7].

The failure of a single fiber link is the most frequent cause of network failure and may cause all lightpaths using the faulty fiber to fail. One common approach for handling faults is to use protection based schemes [6] - [11]. In this approach a primary path and an edge-disjoint backup path are allocated for each connection. The resources for the backup path are allocated at design time, before any failure has occurred. In the absence of failures, communication uses the primary lightpaths alone. When there is a failure due to a fiber cut, communication that uses the disrupted primary lightpaths is resumed using the corresponding backup lightpaths. In dedicated path protection, the backup lightpath is reserved for a single primary lightpath. The resources allocated to this path cannot be used or shared by any other (backup) lightpath. In shared path protection (also called backup multiplexing) resources may be shared by two or more backup paths if and only if the corresponding primary paths are edge disjoint.

III. CENTRALIZED ALGORITHM

We assume that there is a designated node which acts as a control node and all connection requests are processed through this node. We consider a physical fiber network with a set of nodes N and a set of (bi-directional) edges E , where $|N| = N$ and $|E| = E$. Each fiber is capable of carrying a set of K wavelengths, where $|K| = K$. We use Dijkstra's shortest path algorithm [13] to pre-compute a number (R) of edge-disjoint routes, numbered $0, 1, \dots, R-1$, over the physical topology, between every pair of nodes. If we need to establish a connection from source s to destination d , the associated primary and backup lightpaths will use two distinct routes from these R pre-computed routes between s and d . Since the R routes, for a given source-destination pair, are all edge-disjoint, selecting two distinct routes guarantees that the primary and backup lightpaths will be edge-disjoint.

The control node has access to following global information about the network:

- $\rho_{sd}^r = s \rightarrow x_1 \rightarrow x_2 \rightarrow \dots \rightarrow x_{m-1} \rightarrow d$, the r^{th} pre-computed route from source s to destination d , $\forall s, d \in N$, $0 \leq r < R$.
- Λ_e , the set of available channels (wavelengths)

on each edge $e \in E$ in the network.

- The physical route and assigned channel for each lightpath currently established over the network

When a new connection is established, the control node updates its information accordingly. The centralized algorithm (Figure 1) is responsible for allocating resources to the primary and backup lightpath corresponding to each new traffic request, and for updating global information in the control node each time a connection is established or torn down.

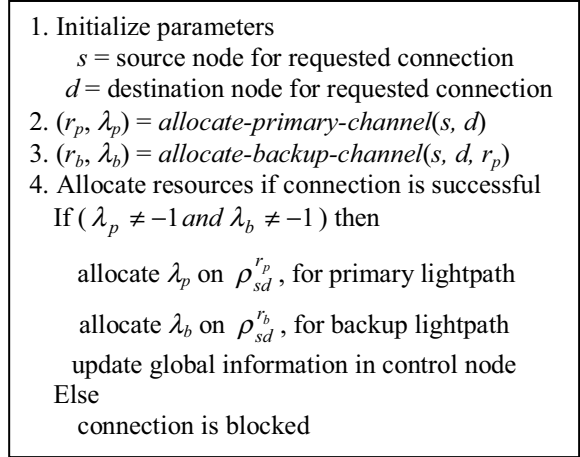


Figure 1. Overview of Centralized Algorithm

The function *allocate-primary-channel* (Figure 2) is responsible for allocating a route and wavelength for the primary lightpath. It looks at each pre-computed route, from s source to destination d , and checks if there is an available wavelength on that route. For the route r , currently under consideration, we calculate the set $L = \bigcap_{e \in \rho_{sd}^r} \Lambda_e$, which is the set of all channels (wavelengths) which are available on each edge e of the r^{th} route i.e. $e \in \rho_{sd}^r$.

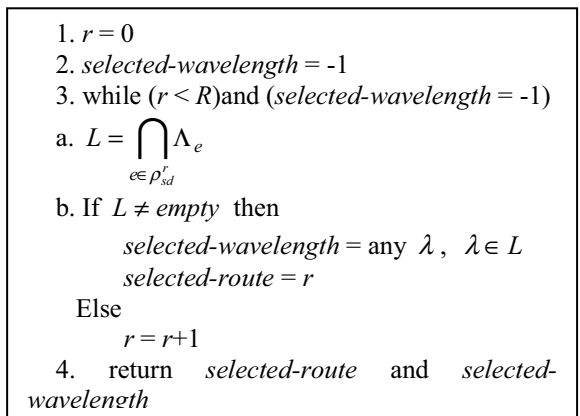


Figure 2. Outline of *allocate-primary-channel* function

If L is empty, it means that there is no single channel that is available on all the edges of the current route.

Based on "Distributed dynamic lightpath allocation in survivable WDM networks", by A. Jaekel, and Y. Chen, which appeared in the Proceedings of International Workshop on Distributed Computing 2005, Kharagpur, India, December 2005. © 2005 Springer.

Therefore, a lightpath cannot be established over that route. If $L = \text{empty}$ for all routes $\rho_{sd}^r, 0 \leq r < R$, then the connection must be blocked and the function returns *selected-wavelength* = -1, indicating failure. If L is not empty for the r th route, then any wavelength $\lambda \in L$ may be selected and a lightpath established over the route. The selected route and channel is returned to the main function, which is responsible for updating network status information

For backup channel assignment, the overall algorithm is very similar to the allocate-primary-channel function. The main difference is that step 3 must be modified, as indicated in Figure 3, and needs to know which route (r_p) was used for the primary lightpath.

```

1.  $r = 0$ 
2. selected-wavelength = -1
3. while ( $r < R$ ) and (selected-wavelength = -1)
  a.  $L = \bigcap_{e \in \rho_{sd}^r} (\Lambda_e \cup \Lambda_{b,e})$ 
  b. If ( $L \neq \text{empty}$ ) and ( $r \neq r_p$ ) then
      selected-wavelength = any  $\lambda, \lambda \in L$ 
      selected-route =  $r$ 
  Else
       $r = r + 1$ 
4. return selected-route and selected-wavelength
    
```

Figure 3. Outline of *allocate-backup-channel* function.

To check if backup multiplexing is possible on an edge e , we calculate the set $\Lambda_{b,e}$. For dedicated protection $\Lambda_{b,e} = 0$, by definition. For shared protection, $\Lambda_{b,e}$ gives the set of wavelengths on edge e that are currently allocated to backup paths, whose corresponding primary paths are edge-disjoint with $\rho_{sd}^{r_p}$. The set of usable channels L can now be calculated as $L = \bigcap_{e \in \rho_{sd}^r} (\Lambda_e \cup \Lambda_{b,e})$ (step 3a). To ensure that the primary and backup lightpaths use two distinct routes, the condition $r \neq r_p$ is included in step 3b.

IV. DISTRIBUTED ALGORITHM

The centralized algorithm requires a single control node that stores information about the state of the entire networks as well as information about every connection that is currently active in the network. The control node must also process and allocate resources for every new connection request. Clearly, such a node can quickly become a bottleneck, as the network size and traffic load increases.

In the distributed scheme, a node does not have global knowledge of the state of the entire network, but operates based only on “local” information. In this

scheme a single node need not know the routes between all source destination pairs, only the routes from itself to other nodes. Similarly, it is not aware of all connections established over the network, or the state of all channels on each edge of the network. It only knows about those connections that are routed through it and those edges that are directly connected to it. Each node stores two main types of information:

i) Network Information: This includes information about the (partial) network topology as well as information about the state of the outgoing links from the node. The link-state information includes two parameters CurrentState (λ) and NumLP(λ) for each channel λ , on each outgoing edge. The network information consists of the following five fields:

- its own node_id
- node_ids of its adjacent nodes and the outgoing link to be used to connect to each adjacent node.
- a set of R edge-disjoint routes from itself to all other nodes in the network.
- CurrentState (λ) for each channel on each outgoing link
- NumLP(λ) for each channel on each outgoing link (required for shared protection only)

CurrentState (λ) for a channel on a particular link refers to one of four possible states –

a) CurrentState (λ) = 0: This indicates that the channel is “free” and is available for allocation to a new lightpath on the link. It has not been assigned to an established lightpath and is also not being considered for a new connection request.

a) CurrentState (λ) = 1: This indicates that the channel is “busy” and has already been allocated for a primary or backup lightpath, on that particular link.

b) CurrentState (λ) = 2: This indicates that the channel was free and is now being considered as a *potential* candidate for allocation to a new lightpath, which is currently in the setup phase. There is a “lock” on the channel, as it is temporarily reserved for the new lightpath. However, the selection has not been finalized, and it is possible that the specified channel will not be selected. If selected, the value of CurrentState (λ) will be set to 1, otherwise it will be reset back to 0

d) CurrentState (λ) = 3: This indicates that the channel was already assigned to one or more backup lightpaths and is now being considered as a *potential* candidate for allocation to another backup lightpath, which satisfies the criteria for backup multiplexing with the other lightpaths using that channel. There is a “lock” on the channel, as it is temporarily reserved for the new (backup) lightpath. However, the selection has not been finalized, and it is possible that the specified channel will not be selected. In either case, the value of CurrentState (λ) will be set to 1. Clearly, this state is needed only for shared path protection, where it is necessary to distinguish between a lock put on a “free” channel and a lock which requires multiplexing with a previously established lightpath.

The value of $\text{NumLP}(\lambda)$, on a particular link, specifies the number of lightpaths that have been assigned to channel λ , on that link. This information is only needed in shared path protection, where more than one backup lightpath may be assigned to the same channel on a given link. For dedicated protection, the value of $\text{NumLP}(\lambda)$ is always 0 (1) if $\text{CurrentState}(\lambda)$ is 0(1).

Lightpath information: Each node in the network stores certain information for each lightpath that is routed through it, whether it is already established or currently being setup.

Each node x_i in the physical route of a lightpath, from a source node s to a destination node d , stores a record called *LP-Record* corresponding to the lightpath. An *LP-Record* consists of seven fields, containing the following information about the lightpath.

- Source node_id
- Destination node_id
- Connection number
- Physical route
- Lightpath Type (primary or backup)
- Selected wavelength (-1 indicates a channel has not yet been assigned)
- Locked channels (set of channels L_i that have been temporarily reserved on edge $x_i \rightarrow x_{i+1}$ for this lightpath)

A. Control Messages

In the distributed approach, each node works independently of the other nodes. Inter-node communication and co-ordination takes place by passing control messages between nodes. These control messages always contain the complete LP-Record of the corresponding lightpath. We assume that the control messages are sent *out of band*, and do not use the same channels as the data traffic. The messages are processed at each node in the physical route and appropriate actions are taken. This includes setting/releasing locks on specific channels, updating the LP-Record, allocating channels to lightpaths and freeing resources when they are no longer being used. We have tried to make the processing steps general so that they can be used for both dedicated and shared path protection with little or no modification. There are four different types of control messages. In this section we will discuss in detail how they are used for dedicated protection. In the next section, we will describe how they should be modified for shared protection.

1) InitiateConnectionSetup

In our scheme a request for a new connection is generated at any given time, based on a predetermined probability p_i ($0 < p_i < 1$). When a connection request is generated, a source node s and a destination node d are selected randomly and an *InitiateConnectionSetup* message is added to the message queue of the selected source node. This type of message is processed at node s ,

which performs the following actions, as shown in Figure 4.

The first step is to assign a unique identifier (C) to the new request. We note that C may not be globally unique, since s is not aware of all connections established over the network. However, C is unique with respect to all connections which originate at node s . Thus, the combination (s, C) can be used to uniquely identify a connection in the entire network. Steps 3 and 4 select the primary and backup routes. The idea here is to choose a route that has a large number free channels on the first edge, with the expectation that this will increase the chances of success. Of course, this is not necessarily the best choice, because other edges in the route may be congested and may not have available channels. But, since we are operating based on local information only, it is a “reasonable” choice. If steps 3 and 4 are successful, the connection setup phase is started and a LP-Record for each lightpath (primary and backup) is created at the source node, with the following information.

- Source node_id = s
- Destination node_id = d
- Connection number = C
- Physical route = $\rho_{sd}^r = s(=x_0) \rightarrow x_1 \rightarrow x_2 \rightarrow \dots \rightarrow x_{m-1} \rightarrow d(=x_m)$
- Lightpath Type = $p(b)$ for the primary (backup) lightpath
- Selected wavelength = -1
- Locked channels = L_0

1. Assign a unique connection_id (C) to the new request
2. $\Lambda_r =$ set of “free” channels on the first edge of ρ_{sd}^r .
3. If $(|\Lambda_{r_p}| = \max\{|\Lambda_r|\}, 0 \leq r \leq R-1 \text{ and } \Lambda_{r_p} \neq \text{empty})$ then
Select $\rho_{sd}^{r_p}$ as the primary route
4. If $(|\Lambda_{r_b}| = \max\{|\Lambda_r|\}, 0 \leq r \leq R-1, r_b \neq r_p \text{ and } \Lambda_{r_b} \neq \text{empty})$ then
Select $\rho_{sd}^{r_b}$ as the backup route
5. If (routes for primary and backup lightpaths selected successfully) then
 - Create a new entry for the connection request
 - Create LP-Records for primary and backup lightpaths
 - Set locks on up to *MaxLocks* wavelengths for each lightpath
 - Send *ForwardRequest* message to next node of each lightpath

Figure 4.

Steps for processing *InitiateConnectionRequest*

For each lightpath, the set of locked channels (L_0) on the first edge is calculated by selecting a subset of w wavelengths from Λ_{r_p} (Λ_{r_b} for backup lightpath), where $w = \min\{|\Lambda_{r_p}| \setminus |\Lambda_{r_b}|, MaxLocks\}$. Here $MaxLocks$ is a preset constant, which specifies the maximum number of wavelengths that can be locked by a single lightpath.

2) *ForwardRequest*

A *ForwardRequest* control message is used in the setup phase of a lightpath. It is responsible for forwarding the LP-Record from a node x_{i-1} to the next node x_i , along the selected route. At each intermediate node x_i , the message is processed and the required channels (if available) are reserved on the appropriate outgoing link, before forwarding the LP-Record to the next node (Figure 5)).

1. If ($x_i = d$) then
 - a. Select any $\lambda \in L_{i-1}$
 - b. Update LP-Record fields
 $selected-wavelength = \lambda$
 - c. Send *ResponseSignal* message with updated LP-Record back to x_{i-1}
 2. If ($x_i \neq d$) then
 - If ($L_{i-1} \cap \Lambda_e \neq empty$) then
 - a. $L_i =$ a subset of w channels from $L_{i-1} \cap \Lambda_e$
 - b. Update LP-Record fields
Locked channels = L_i
 - c. Save updated LP-Record
 - d. For all $\lambda \in L_i$ on edge $e = x_i \rightarrow x_{i+1}$
CurrentState(λ) = 2 (i.e. locked)
 - e. Send *ForwardRequest* message with updated LP-Record to x_{i+1}
 - Else
 - f. Update LP-Record fields
 $selected-wavelength = -1$
 - g. Send *ResponseSignal* message with updated LP-Record back to x_{i-1}

Figure 5. Steps for processing *ForwardRequest* control message

If x_i is the destination node, it means that there is at least one free channel along the entire route. In this case, x_i , selects a wavelength $\lambda \in L_{i-1}$ and sends a *ResponseSignal* message back to x_{i-1} , indicating that the request was successful. If x_i is not the destination, it checks if for available channels on the appropriate outgoing edge, before forwarding the request to the next node.

If x_i is an intermediate node, then L_{i-1} gives the set of usable channels for the lightpath up to node x_i and $L_{i-1} \cap \Lambda_e$, where $e = x_i \rightarrow x_{i+1}$ is the set of usable channels on the outgoing link e . If L_i is empty, then there are currently no usable channels available on edge e and a response signal, indicating failure, is sent to node x_{i-1} .

If $L_{i-1} \cap \Lambda_e \neq empty$, then the w channels of the selected subset L_i are “locked” on edge e (Step 2d in Figure 5), for the current lightpath, and the LP-Record is updated so that Locked channels = L_i . As mentioned before, the number of locked channels (w) is the lower of the predefined constant $MaxLocks$ and the actual number of available channels on edge $e = x_i \rightarrow x_{i+1}$. Finally, the updated LP-Record is sent to node x_{i+1} in a *ForwardRequest* control message.

3) *ResponseSignal* message

This type of message is sent “backwards” along the physical route of a lightpath. It indicates either i) the lightpath setup request failed ($selected-wavelength = -1$) or ii) a suitable channel λ_s was available along the route and has been assigned to the lightpath ($selected-wavelength = \lambda_s$). When a node x_i receives a *ResponseSignal* message, it releases the locks on all wavelengths $\lambda \in L_i, \lambda \neq \lambda_s$ (step 2c), and updates the local LP-Record and status information of the relevant channels on edge $e = x_i \rightarrow x_{i+1}$. Then the *ResponseSignal* message is sent to node x_{i-1} . An outline of the required steps is given in Figure 6.

- Check local copy of LP-Record to determine L_i

 1. If ($selected-wavelength == -1$) then
 - a. For all $\lambda \in L_i$ on edge $e = x_i \rightarrow x_{i+1}$
CurrentState(λ) = CurrentState(λ) - 2
 - b. If ($x_i \neq s$) Send *ResponseSignal* to x_{i-1}
 - c. Delete local copy of LP-Record
 2. Else if ($selected-wavelength == \lambda_s$) then
 - a. Update local copy of LP-Record
 $selected-wavelength = \lambda_s$
 - b. CurrentState(λ_s) = 1(i.e. Busy) on edge $e = x_i \rightarrow x_{i+1}$
 - c. NumLP(λ_s) = NumLP(λ_s)+1 on edge $e = x_i \rightarrow x_{i+1}$
 - d. For all $\lambda \in L_i, \lambda \neq \lambda_s$ on edge $e = x_i \rightarrow x_{i+1}$
CurrentState(λ) = CurrentState(λ) - 2
 - e. If ($x_i \neq s$) Send *ResponseSignal* to x_{i-1}

Figure 6. Steps for processing *ResponseSignal* message

Although it has not been included in Figure 6, there is some additional processing that must be done, when a *ResponseSignal* message is received at the source node s . We know that there are two lightpaths (primary and

backup) for each connection request. When the source node receives a *ResponseSignal* for one lightpath, it checks if it has already received a response for the other lightpath. If not, it simply waits until both responses are available. Once responses for both lightpaths (primary and backup) have been received, we need to consider three possibilities.

Case 1: Both responses are successful: In this case, the connection request is successful and communication can begin along primary path.

Case 2 (Both responses indicate failure): In this case, the connection is blocked and the corresponding entry is deleted from the node

Case 3 (One indicates success, the other failure): In this case, the connection is also blocked, but the resources allocated to the successful lightpath must be reclaimed. A *FreeResources* control message containing the LP-Record of the successful lightpath, is sent the next node(x_1) of the route ρ_{sd}^r . Then the local copy of the LP-Record (for the successful lightpath) and the entry corresponding to the connection are both deleted from the node.

4) *FreeResources*

This type of message is used to reclaim resources allocated to a lightpath, when they are no longer needed. A *FreeResources* message is generated at the source node of a connection for one of two reasons: a) a successfully established connection is needs to be terminated and the corresponding resources reclaimed b) one of the lightpaths for a new connection request was successfully established, but the other failed (Case 3 in section 4.1.3).

1. $\lambda_s = \text{selected-wavelength}$
2. If ($x_i \neq d$) then
 - a. $\text{NumLP}(\lambda_s) = \text{NumLP}(\lambda_s) - 1$ on edge $e = x_i \rightarrow x_{i+1}$
 - b. If ($\text{NumLP}(\lambda_s) = 0$) then
 $\text{CurrentState}(\lambda_s) = 0$ (i.e. Free) on edge $e = x_i \rightarrow x_{i+1}$
 - c. Send *FreeResources* control message to x_{i+1}
3. Delete local copy of LP-Record

Figure 7. Steps for processing *FreeResources* message

A node x_i processes a *FreeResources* message by releasing the channel allocated to the lightpath on its outgoing link. If there are no other lightpaths assigned the same channel on edge $e = x_i \rightarrow x_{i+1}$ (i.e. $\text{NumLP}(\lambda_s) = 0$), then the current state of λ_s is reset to 0. This will always be the case for dedicated path protection, or if the lightpath being considered is a primary lightpath. However, for shared protection, it is possible the more than one backup lightpath is assigned the same channel λ_s on the same edge. In that case, $\text{NumLP}(\lambda_s) > 0$, even after the lightpath is removed, and

$\text{CurrentState}(\lambda_s)$ should remain at 1, and not be reset to 0. This is accomplished in step 2b of the process. Finally, the *FreeResources* message is sent to the next node x_{i+1} on the physical route for the lightpath and the local copy of the LP-Record is deleted from the node. An outline of the process is given in Figure 7.

B. Shared Path Protection

In shared path protection, two backup lightpaths can be assigned the same channel on an edge e , if the corresponding primary routes are edge disjoint. This means that in order to determine if backup multiplexing can take place, a node must have information about the physical route for the corresponding primary lightpath as well. In order to meet this requirement, we augment the information in the LP-Record. Instead of having a single Physical route field, we have two fields called Primary Route and Backup Route. Therefore, each LP-Record contains the following eight fields, instead of seven.

- Source node_id = s
- Destination node_id = d
- Connection number = C
- Primary route
- Backup route
- Lightpath Type = $p(b)$ for the primary (backup) lightpath
- Selected wavelength (-1 indicates a channel has not yet been assigned)
- Locked channels

The distributed scheme for shared path protection uses the same framework as the dedicated scheme. Resources are allocated and released, using the four types of control messages. The *InitiateConnectionRequest*, *ResponseSignal*, and *FreeResources* messages are processed in the same way as for dedicated protection. Only the processing of *ForwardRequest* messages need to be modified slightly and will be discussed in this section.

5) Modification of *ForwardRequest* message

For shared path protection, the process for reserving usable channels on the outgoing edge (step 2 of Figure 5) needs to be changed. Since, the rest of the steps are the same, we will only discuss step 2 in detail here. An outline of this step is given below in Figure 8.

In this case, the set of usable channels on an edge e , includes not only the “free” channels (Λ_e), but also those channels in $\Lambda_{b,e}$. If the current lightpath is a primary lightpath, then by definition $\Lambda_{b,e} = \text{empty}$. Otherwise, $\Lambda_{b,e}$ gives the set of channels λ on edge e , such that:

- i) λ has already been allocated to one or more backup lightpaths and
- ii) all the corresponding primary lightpaths are edge-disjoint with respect to the primary route of the current lightpath.

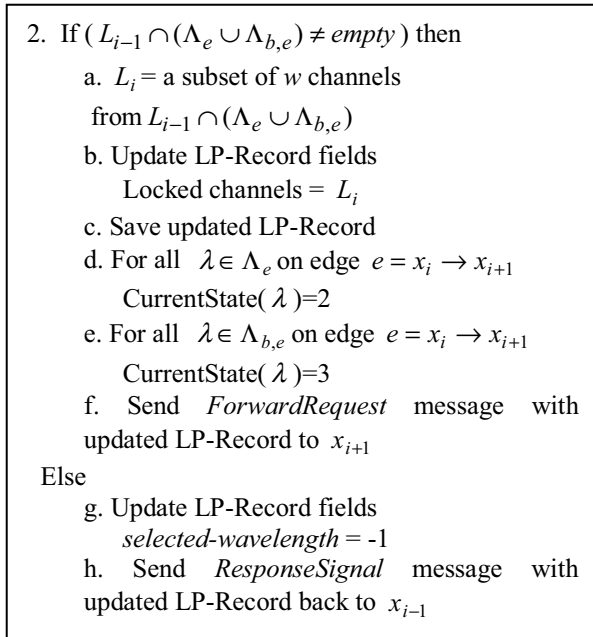


Figure 8. *ForwardRequest* message for shared protection

Since each node stores the complete LP-Record (including primary and backup routes) of all lightpaths routed through it, it is not difficult to calculate $\Lambda_{b,e}$. When we “lock” the channels in L_i , it is necessary to distinguish between the channels in the set Λ_e and those in $\Lambda_{b,e}$. This is because, a channel $\lambda_b \in \Lambda_{b,e}$ has previously been assigned to another backup lightpath, and its initial state (before being locked) is given by CurrentState(λ_b)=1, whereas for $\lambda \in \Lambda_e$, its initial state (before being locked) is given by CurrentState(λ)=0. Therefore, when releasing the locks on the channels, we need to set them back to the *correct* initial state. When a *ResponseSignal* message is processed, step 2d does exactly this. For dedicated protection, when a channel is locked, its state is always set to 2. Therefore, the steps given in Figure 6, can be used for both dedicated and shared protection.

V. EXPERIMENTAL RESULTS

In this section, we will evaluate the performance of the two algorithms we have presented. We will look at both the solution time and the quality of the solutions, by comparing them with “optimal” solutions generated from exact ILP formulations [5]. Figure 9 shows the total number of connections that can be accommodated by the network for (a) dedicated protection and (b) shared protection. We see that the performance of the centralized algorithm is very close to the optimal. As expected, more connections are blocked with the distributed algorithm.

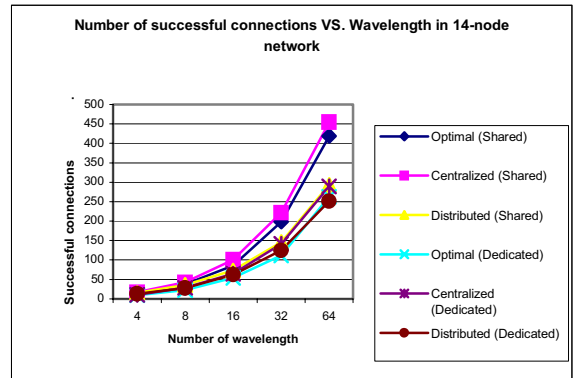


Figure 9. Performance of our heuristics compared to the optimal solutions

This is expected since,

- i) In the centralized approach (this includes optimal ILP formulations) connection requests are presented to the control node sequentially. But in the distributed approach several connections may be in the setup phase simultaneously. This means many channels could be “reserved” by one lightpath, and hence cannot be considered by another lightpath, even if they are ultimately released.

- ii) In the centralized approach, all possible physical routes are considered for a lightpath from s to d . In the distributed approach, we pre-select a specific route for a lightpath (based on incomplete local information only). This can reduce the chances of success.

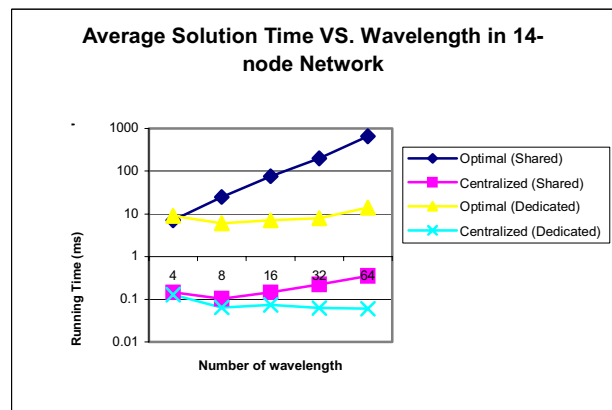


Figure 10. Solution time for heuristics compared to ILPs

Figure 10 compares the solution times for the ILP and centralized heuristic. We see that the solution time for ILP increases very rapidly and it quickly becomes infeasible. The centralized algorithm is several orders of magnitude faster. Even for dedicated protection. Therefore, our algorithms are useful for practical networks, where ILPs cannot be used.

For the distributed algorithm, the performance of the algorithm depends on the parameter *MaxLocks*, which specifies the maximum number of channels that can be reserved for a single lightpath. It is important to choose the value of *MaxLocks* appropriately. If the value is too small, it will reduce the chances of success for the given

lightpath. If the value is too high, too many wavelengths are locked unnecessarily and it becomes difficult for other lightpaths to find free channels. We have experimented with several different values, from $MaxLocks = 1$ to $MaxLocks = K$ (the maximum number of channels per fiber). Figure 11 shows how the performance of the algorithm changes with the value of $MaxLocks$. Initially, the performance improves, since the chances of success for a given lightpath increases if more channels are reserved for it. However, after a certain point, the performance starts decreasing as $MaxLocks$ is increased further. This is because, too many channels are becoming locked by a single lightpath, reducing the chances of success for other lightpaths (also in the setup phase) that may try to use the same link.

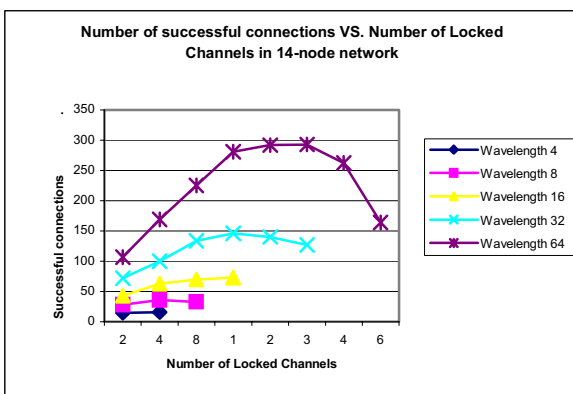


Figure 11. Variation of performance with the number of locked channels

VI. CONCLUSIONS

In this paper we have presented two algorithms for dynamic lightpath allocation in survivable WDM networks. Our algorithms automatically allocate both a primary and a backup lightpath for each connection request, in a way that tries to minimize the amount or resources required. We have compared our algorithms with optimal solutions, generated from ILP formulations. The results indicate that the performance of our *centralized* algorithm is always within 90% of the optimal, while the required running time is less than 1% of the ILP running times (for both dedicated and shared protection). In the second approach, we sacrifice some performance in order to make the algorithm distributed. In this scheme the network nodes can operate independently, based only on local information, and communicate by passing control messages. The simulation results demonstrate that this is a viable and attractive option for practical networks. Although ILP formulations provide interesting theoretical insights, they are computationally intractable for practical networks, which typically require real-time resource allocation. The algorithms presented in this paper produce solutions comparable to the optimal solutions and can be used even in large networks.

ACKNOWLEDGMENT

The work of A. Jaekel was supported by research grants from the Natural Science and Engineering Research Council (NSERC), Canada.

REFERENCES

- [1] T.E. Stern, K. Bala, "Multiwavelength optical networks - a layered approach" (Addison Wesley, Inc., 1999).
- [2] I. Chlamtac et al., "Lightnets: topologies for high-speed optical networks," IEEE/OSA Journal of Lightwave Technology, vol. 11, (May/June, 1993), pp. 951-961.
- [3] O. Gerstel et al., "Dynamic channel assignment for WDM optical Networks with little or no wavelength conversion," Proc. 34th Annual Allerton Conf (Monticello, IL, Oct. 1996), pp. 32-43.
- [4] 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, January 2000.
- [5] Y. P. Aneja, S. Bandyopadhyay and A. Jaekel, "An Efficient Protection Scheme for WDM Networks Using Dynamic Lightpath Allocation", in *HPC Asia02*, pp. 269-274, Dec. 2002.
- [6] C. Ou, J. Zhang, H. Zang, L. Sahasrabudhe, and B. Mukherjee, "New and Improved Approaches for Shared-Path Protection in WDM Mesh Networks," IEEE/OSA J. of Lightwave Tech., Vol. 22, No. (2004) 1223-1232.
- [7] S. Zhong and A. Jaekel, "Optimal Priority Based Lightpath Allocation for Survivable WDM Networks," Int. Conf. on Computers, Communications and Networks (2004), pp. 17-22.
- [8] Ramaswami and B. Mukherjee, "Survivable WDM Mesh Networks, Part 1-Protection", Proc. of IEEE INFOCOM'99, pp. 744-751, March, 1999.
- [9] L. Sahasrabudhe, S. Ramamurthy and B. Mukherjee, "Fault management in IP-over-WDM networks: WDM protection versus IP restoration" IEEE J. Select. Areas Commun., Vol. 20, No. 1, Pages 21 - 33. Jan. 2002.
- [10] S. Ramamurthy, L. Sahasrabudhe and B. Mukherjee, "Survivable WDM mesh networks", J. of Lightwave Technology, vol. 21, no. 4, pp. 870-883, April 2003.
- [11] M. Sridharan, M.V. Salapaka, A..Somani, "A practical approach to operating survivable WDM networks," IEEE JSAC, Vol. 20, No. 1, Jan. 2002 pp. 34-46.
- [12] H. Zang, C. Ou and B. Mukherjee, "Path-protection RWA in WDM mesh networks under duct-layer constraints", IEEE/ACM transactions on networking, vol. 11, no. 2, pp. 248-258, April 2003.
- [13] E. Dijkstra, "A note on Two problems in connection with graphs", Numerical mathematics:1, pp269-271, 1959.

Arunita Jaekel obtained her B. Engg. degree in electronics and telecommunications engineering from Jadavpur University, India (1988). She received her M. A. Sc. (1990) and Ph.D. (1995) in electrical engineering from University of Windsor, Canada. She has been with the School of Computer Science at University of Windsor since 1995, where she is currently an associate professor. Her research interests include optical communications and wireless sensor networks.

Ying Chen is currently pursuing her Bachelors degree in computer science at the University of Windsor, Canada.