Preserving Survivability During Logical Topology Reconfiguration in WDM Ring Networks

[1]Hwajung Lee, [1]Hongsik Choi, [2]Suresh Subramaniam, and [1]Hyeong-Ah Choi
Department of [1]Computer Science and [2]Electrical and Computer Engineering
The George Washington University
Washington, DC 20052
{hjlee,hongsik,suresh,choi}@seas.gwu.edu

Abstract

We consider the design of reconfiguring logical topologies over physical WDM ring networks. The logical topology consists of the same set of nodes as the physical topology, and the links of the logical topology are lightpaths established (or embedded) over the physical topology. The logical topology is said survivable if the failure of any single physical link does not disconnect the logical topology. In this paper, we consider the following problem. Given a logical topology with its survivable embedding over a physical ring network and a new logical topology to be reconfigured, find a sequence of lightpaths additions and deletions satisfying the given wavelengths and ports constraints such that the logical topology remains survivable throughout the reconfiguration.

Keywords: Network Survivability, Wavelength Constraint, Reconfiguration Cost.

1 Introduction

Optical networks employing Wavelength Division Multiplexing (WDM) and wavelength-routing are capable of providing lightpaths to higher service layers. Lightpaths are optical circuit-switched paths that have transmission rates of a few Gb/s. By the use of WDM, multiple lightpaths may traverse the same optical fiber link, each one using a different wavelength.

Survivability is a critical requirement for high-speed optical networks. There has been a large amount of work that focuses on pre-allocating backup capacity so that any failed lightpaths may be restored rapidly as soon as normal operation is disrupted in the event of link break. The proposed techniques are classified as either link protection or path protection, depending on whether the rerouting of lightpaths is done around the failed link, or on an end-to-end basis. Protection at the optical layer is considered to be fast, partly because of the proximity of the optical layer to the physical layer at which the failure is first detected, and partly because of the coarse granularity at which restoration is done (at the lightpath or fiber level).

When an electronic service layer is embedded over a WDM optical network, then it may be the case that the electronic layer incorporates its own survivability functions, thereby making the optical layer recovery redundant, and in the worst case, perhaps conflicting. Furthermore, when a physical link fails, it may not be necessary for all the affected lightpath traffic to be restored. Thus, there is a case to be made for recovery to be done solely at the electronic layer. If the electronic layer were the IP layer, then the only requirement for the layer to be survivable is that it be connected.

Motivated by the above, we have considered in [2] the embedding of an electronic layer on a physical WDM network such that the electronic layer network is connected when a single link fails. The connectivity at the electronic layer is represented by the *logical topology*. The logical topology is a topology which has as its nodes the set of electronic nodes. The edges of the logical topology correspond to the set of lightpaths that are established over the physical topology. As mentioned above, multiple lightpaths may be routed over the same physical link, and therefore, it is possible for a single physical link failure to break more than one edge on the logical topology. Since survivability at the logical topology depends on the availability of multiple routes between nodes at the logical layer, it is clear that there must be some amount of coordination between the two layers if survivability has to be achieved at the logical layer. In [2], we focussed on the design of logical topologies that are *survivable*. We defined a logical topology to be survivable if the failure of any single physical link does not disconnect the logical topology. Survivable logical topology design not only involves the determination of the logical edges but on the embedding of those edges on the physical topology, i.e., the routing of the lightpaths.

Consider a logical topology shown in Figure 1 (a) corresponding to a connection request set $C = \{(0,2),(2,4),(4,0),(1,3),(3,5),(5,1),(0,1),(2,5)\}$ to be embedded over a WDM ring network with six nodes. Figure 1 (b-c) show the physical ring topology and two different lightpaths assignments, in which the logical topology maintains its connectivity in the presence of any single physical link failure when the lightpath setup is done using the routes shown in (b), and it does not when the setup is done using the routes in (c) and when link $(0,1)$ fails.

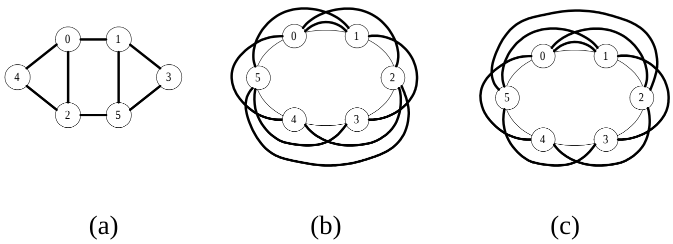


Figure 1. (a) A logical topology, (b) a survivable embedding, and (c) a non-survivable embedding.

There has been other recent research in the design of survivable logical topologies. In [1], the problem of embedding lightpaths such that the minimum number of source-destination pairs are disconnected at the logical layer was considered, and some optimization heuristics were presented. In [3], a similar problem was considered and some conditions for the survivability of a logical topology were presented. In both of these papers, the physical topology was assumed to be an arbitrary mesh.

In this paper, we address the problem of reconfiguring the network from logical topology G_1 to logical topology G_2 in such a way that the logical topology remains connected in the presence of any single physical link failure (i.e., survivable) throughout the reconfiguration process. Our goal is to find a sequence of lightpaths additions (i.e., finding routes and wavelength assignments) and deletions such that the logical topology's survivability is maintained during the entire period of reconfiguration. We consider in this paper a physical ring network. Ring networks are important because the prevalent topology for SONET is the ring. As these networks are upgraded to WDM, it is likely that the topology will be maintained for some time before growing into a mesh network. Secondly, the simplicity of the topology enables us to take a deeper look into the complexity of the problem.

In the next section, we formally state the problem we attempt to solve in this paper. Some insight into the complexity of the problem is presented in Section 3. In Section 4, we present a simple approach for reconfiguration followed by the discussion on a *bad* choice from multiple feasible embeddings. In Section 5, we propose a heuristic algorithm for finding feasible reconfiguration using the minimum *reconfiguration cost* with the objective of minimizing the number of additional wavelengths. Simulation results are also given. Concluding remarks in Section 6 complete the paper.

2 Network Model and Problem Formulation

Let R denote a ring network with n nodes. Each link is bidirectional supporting W wavelengths channels. Each node is assumed to have p ports that can be used as a source or a sink of up to p lightpaths.

Let G_1 and G_2 be logical topologies for R such that G_1 and G_2 both are survivable, i.e., G_1 and G_2 both have survivable embeddings in R. Given a survivable embbeding of G_1 in R corresponding to the current set of lightpaths established over R and a logical topology G_2 corresponding to a new set of lightpaths to be reconfigured from G_1, a reconfiguration process is called *survivable* if during the entire period of reconfiguration,

(i) the logical topology remains survivable (i.e., connected under the failure of any single physical link), and

(ii) the port and wavelengths constraints are satisfied.

Our problem is to find a survivable reconfiguration of the network from G_1 to G_2 by establishing a sequence of lightpaths additions and deletions.

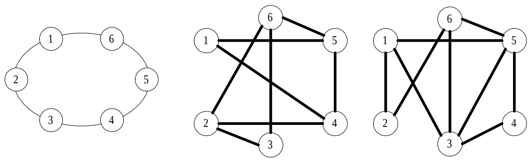
3 Problem Complexity

If there is no constraint on the number of lightpaths that can be established at each node and the number of wavelengths that can be used in each link, one can simply add all lightpaths in $G_2 \setminus G_1$ to G_1 and form $G_1 \cup G_2$, and then delete all lightpaths in $G_1 \setminus G_2$. (Assume that survivable embeddings of G_1 and G_2 are used in the setup of lightpaths.) This will ensure the survivability of the logical topology throughout the reconfiguration process. On the other hand, if the logical topology (including all nodes in R) corresponding to the set of existing lightpaths in $G_1 \cap G_2$ is connected, a survivable reconfiguration can be easily done first by deleting all lightpaths in $G_1 - (G_1 \setminus G_2)$, and then by adding lightpaths in $G_2 \setminus G_1$.

The above observations suggest that reconfiguration steps for adding and deleting lightpaths must be designed carefully to find a feasible solution. In what follows, we illustrate the complication of the problem even further by examining three different cases.

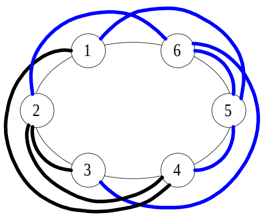
<u>CASE 1 :</u> A feasible solution that modifies the current embedding of some lightpaths in $G_1 \cap G_2$.

Consider a ring network R with $W = 4$ and $p = 4$, and two logical topologies G_1 and G_2 to be embedded over R as shown in Figure 2, where $G_1 \setminus G_2$ = $\{(1,4),(2,3),(2,4)\}$, $G_1 \cap G_2$ = $\{(1,5),(2,6),(3,6),(4,5),(5,6)\}$, and $G_2 \setminus G_1$ = $\{(1,2),(1,3),(3,4),(3,5)\}$. Survivable embeddings of G_1 and G_2 are shown in Figure 3.

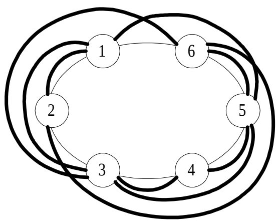


(a) R (b) G_1 (c) G_2

Figure 2. Physical and Logical Topologies



(a) G_1 over R



(b) G_2 over R

Figure 3. Survivable Embeddings

Now, we are going to reconfigure the logical topology from G_1, whose current embedding is shown in Figure 3 (a), to G_2 as shown in Figure 2 (c). Suppose there exists a survivable embedding of G_2 without changing any of the current lightpaths in $G_1 \cap G_2$. The current lightpaths in $G_1 \cap G_2$ are shown in Figure 4. Consider the two lightpaths (corresponding to logical links (2,1) and (2,6) in G_2) to be established at node 2. If the lightpath connecting 2 and 6 is kept as it is in Figure 3 (a), the lightpath connecting 2 and 1 has to be established in the counter-clockwise direction from 2 to 1 since otherwise the failure of physical link $(1,2)$ will isolate node 2 from the remaining network. However, in such an embedding, if the physical link $(1,6)$ fails, both lightpaths corresponding to logical links $(2,1)$ and $(2,6)$ will fail and the failure of these two lightpaths will make again node 2 isolated from the rest of the network. Therefore, any feasible solution must modify the current embedding of the lightpath between 2 and 6 (i.e., a lightpath between 2 and 6 must be re-established in the counter-clockwise direction from 2 to 6). The embedding shown in Figure 3 (b) is such an embedding.

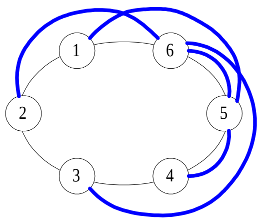


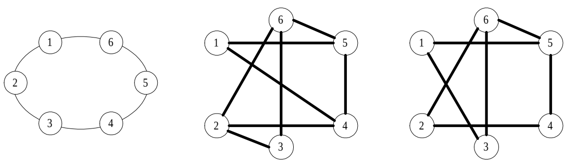
Figure 4. $G_1 \cap G_2$

<u>CASE 2 :</u> A feasible solution that temporarily deletes and reestablishes some lightpaths in $G_1 \cap G_2$ due to the wavelength constraint.

Consider a ring network R of 6 nodes with $W = 3$ and $p = 4$ and two logical topologies G_1 and G_2 to be embedded over R as shown in Figure 5, where $G_1 \setminus G_2 = \{(1,4),(2,3)\}$, $G_1 \cap G_2 = \{(1,5),(2,4),(2,6),(3,6),(4,5),(5,6)\}$, and $G_2 \setminus G_1 = \{(1,3)\}$. Survivable embeddings of G_1 and G_2 are shown in Figure 6.

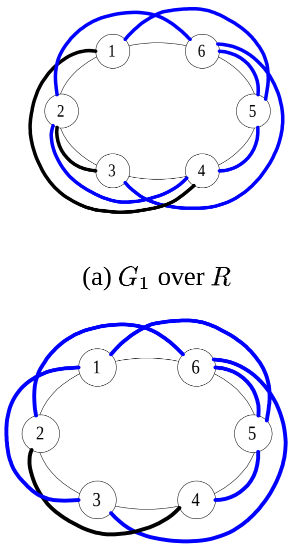
Note that during the reconfiguration, lightpaths corresponding to logical links $(1,4)$ and $(2,3)$ must be deleted and a new lightpath corresponding to logical link $(1,3)$ must be established. Suppose there is a feasible solution that only adds lightpaths in $G_2 \setminus G_1$ and delete lightpaths in $G_1 \setminus G_2$ during the entire reconfiguration process. If lightpath $(1,4)$ or $(2,3)$ is deleted before adding lightpath $(1,3)$, the failure of physical link $(1,6)$ or $(3,4)$, respectively, will make node 1 or 3 isolated. Hence, any feasible solution in this case must add lightpath $(1,3)$ before deleting lightpath $(1,4)$ or $(2,3)$. So we consider two cases for the setup of lightpath between 1 and 3: clockwise from 1 to 3 and counter-clockwise from 1 to 3.

If lightpath between 1 and 3 is added in the counter-clockwise direction from 1 to 3, then the existing lightpath $(2, 4) \in G_1 \cap G_2$ must be deleted beforehand since otherwise four lightpaths will use physical link $(2, 3)$, violating the wavelength constraint. Now assume that lightpath between 1 and 3 is added in the clockwise direction from 1 to 3. Similarly, one of the existing lightpaths $(2, 4), (3, 6) \in G_1 \cap G_2$ must be deleted before adding lightpath $(1, 3)$ since otherwise the wavelength constraint will be violated on the physical link $(3, 4)$. In either case, at least one existing lightpath in $G_1 \cap G_2$ must be temporarily deleted and reestablished later.



(a) R (b) G_1 (c) G_2

Figure 5. Physical and Logical Topologies



(a) G_1 over R

(b) G_2 over R

Figure 6. Survivable Embeddings

<u>CASE 3 :</u> A feasible solution that temporarily adds some lightpaths not in $G_1 \cup G_2$ to guarantee the survivability during the reconfiguration period.

Consider the same example for the physical and logical topologies discussed for CASE 2 (see Figures 5 and 6). As discussed in CASE 2, any feasible solution cannot delete lightpath $(1, 4)$ or $(2, 3)$ without adding new lightpaths. In what follows, we present a feasible solution by temporarily adding a lightpath that is not in $G_1 \cup G_2$ and deleting it later.

Initially, the lightapths are as shown in Figure 6 (a). A lightpath between 1 and 2 is temporarily added in the counter-clockwise direction from 1 to 2. We then safely delete the existing lightpath $(1, 4)$, and add a new lightpath between 1 and 3 in the counter-clockwise direction from 1 to 3. The existing lightpath $(2, 3)$ is now deleted, and then the temporary lightpath $(1, 2)$ is finally deleted.

4 A Simple Reconfiguration Approach

As discussed in the previous section, maintaining logical topology's survivability during the entire reconfiguration period requires a careful design of lightpaths additions and deletions. But, if the current setup of lightpaths only uses up i to $W - 1$ wavelengths in each of the physical link and up to $p-2$ ports at each node, one can easily find a feasible solution using the following steps: (i) add a lightpath between each pair of adjacent nodes, (ii) delete all lightpaths in G_1, (iii) establish all lightpaths in G_2 based on its survivable embedding, and (iv) delete all lightpahs constructed in (i).

This procedure is simple and may encounter difficulties as discussed in the next section.

4.1 Embedding Choice

Intuitively, the implementation of our simple approach presented in the previous section would be always feasible if the number of wavelengths W and the number of ports p are large and the current logical topology G_1 has only a small number of lightpaths established at each node (except possibly for a few nodes). The following discussion exhibits a construction of a *bad* (yet survivable) embedding of a logical topology that would make its reconfiguration to other logical topology difficult.

Let n denote the number of nodes in a ring physical network R, and $W = n - k + 1$, for any k, $1 \leq k \leq n$, (where W is an arbitrary integer in $1 \leq W \leq n$) denote the number of wavelengths supported by each link in R. It is assumed that the number of ports p available at each node is equal to $2W$. Hence, the wavelength (not the port) availability is a major constraint to be considered in the establishment of a new lightpath. Figure 7 shows a survivable embedding of a logical topology G over a ring. The set of logical links in G is given as $\{(n, i), (i, n - k) \mid 1 \leq i \leq n - k - 1\}$

$\cup\{(j, j+1) \mid n-k \leq j \leq n-1\}$, and the route of each lightpath corresponding to each logical link is as shown in Figure 7. Note that the number of lightpaths established in each node, except for nodes n and $n-k$, is only 2. However, each link between n and $n-k$ in the counter-clockwise direction has fully utilized its available wavelengths (i.e., $n-k+1$). Therefore, implementing our simple algorithm suggested in Section 4 would be impossible.

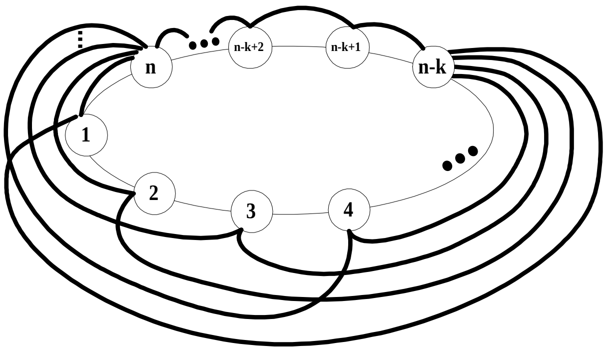


Figure 7. $W = n - k + 1$**.**

The above discussion suggest that the choice of a survivable embedding, when there are multiple choices, is important for survivable reconfiguration.

5 Reconfiguration Algorithm using the Minimum Reconfiguration Cost

The *reconfiguration cost* is defined based on the total number of lightpaths added and the total number of lightpaths deleted to reconfigure from one embedding M_1 to another embedding M_2. Let α denote the cost to establish one lightpath and β denote the cost to delete one lightpath. Then, the total reconfiguration cost is

$$COST \geq \alpha \cdot A + \beta \cdot D,$$

where $A = |M_2 - M_1|$ and $D = |M_1 - M_2|$.

As discussed earlier, if unlimited number of wavelengths is available, one can simply add all lightpaths in $M_2 - M_1$, and then delete all lightpaths in $M_1 - M_2$. This clearly can be done using the minimum reconfiguration cost. Note that the total number of wavelengths used in reconfiguration is $W_{reconf} \geq \max\{W_{M_1}, W_{M_2}\}$, where W_{M_1} and W_{M_2} denote the numbers of wavelengths used in embeddings M_1 and M_2.

In this section, we propose a heuristic algorithm to minimize W_{reconf} while the reconfiguration cost is preserved minimum. Note that no temporary lightpath will be added or deleted in our heuristic algorithm to maintain the minimum reconfiguration cost. Our heuristic algorithm is presented in Algorithm *MinCostReconfiguration*, which takes M_1, M_2, and G_P as input and return W_{add} as an output, where M_1 and M_2 denote survivable embeddings of current and new logical topologies, G_P denote the physical topology, and $W_{add} = W_{reconf} - \max\{W_{M_1}, W_{M_2}\}$.

5.1 Numerical Results

In our simulation, the number of nodes in the ring is $n = 8$, 16, and 32. Logical topologies are randomly genereated using the edge density 50%. Let G_1 and G_2 be two logical topologies reconfigured from G_1 to G_2. We define *difference factor* to be the number of logical edged in $G_1 - G_2$ plus the number of logical edges in $G_2 - G_1$ divided by the maximum possible number of logical edges in a graph with n nodes (i.e., $DiffFactor = 2(|(E(G_1) - E(G_2)| + |E(G_2) - E(G_1)|)/(n(n-1)))$. We consider the difference factor 10%, 20%, $\cdots$, 100%. Each simulation is executed 500 times. In Figure 8, the simulation results show W_{add} for each case.

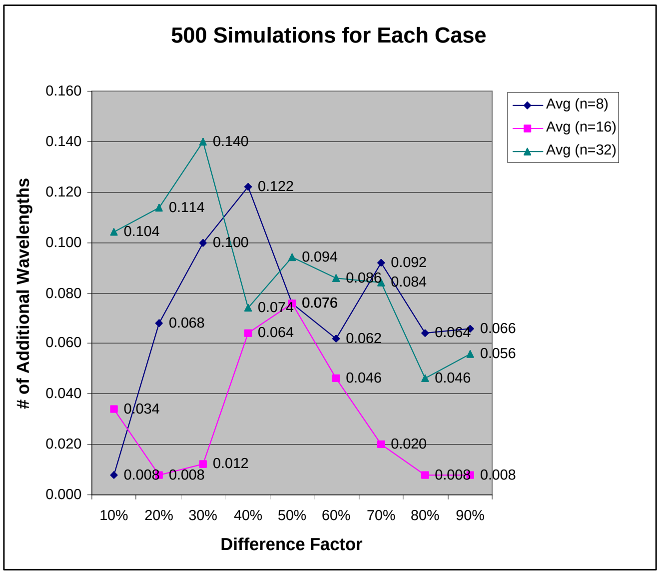


Figure 8. Simulation Results.

Figures 9, 10, 11 show the maximum, minimum, and average numbers of required additional wavelengths, the number of wavelengths for M_1, and the number of wavelengths for M_2. The numbers of different connection requests between the first logical topology and the second logical topology are shown in the column of *# of Diff Conn Req (Simulation)* observed from the simulation results and in the column of *Expected # of Diff Conn Req (Calculated)* calculated based on the number of edges in the logical topology, respectively.

	$\langle W_{ADD} \rangle$			$\langle W_{M1} \rangle$			$\langle W_{M2} \rangle$			# of Diff Conn Req. (from Simulation)	Expected # of Diff Conn Req.(Calculated)
	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg		
10%	1	0	0.008	8	4	5.784	8	3	5.464	1.091	1.400
20%	2	0	0.068	8	3	5.770	7	3	5.388	2.375	2.800
30%	2	0	0.100	8	3	5.692	8	3	5.380	3.762	4.200
40%	2	0	0.122	8	4	5.806	8	3	5.282	5.420	5.600
50%	2	0	0.076	8	4	5.800	8	3	5.368	6.710	7.000
60%	2	0	0.062	8	3	5.796	8	3	5.180	8.212	8.400
70%	2	0	0.092	8	3	5.772	7	3	5.086	9.433	9.800
80%	2	0	0.064	8	3	5.772	8	3	4.850	10.869	11.200
90%	1	0	0.066	8	4	5.750	7	3	4.736	12.099	12.600
			Average	**8**	**3.4**	**5.771**	**7.7**	**3**	**5.193**		

Figure 9. Number of Node = 8.

	$\langle W_{ADD} \rangle$			$\langle W_{M1} \rangle$			$\langle W_{M2} \rangle$			# of Diff Conn Req. (from Simulation)	Expected # of Diff Conn Req.(Calculated)
	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg		
10%	3	0	0.034	21	10	14.588	19	8	13.360	5.971	6.000
20%	1	0	0.008	20	11	14.668	20	7	13.026	12.155	12.000
30%	2	0	0.012	21	9	14.698	20	7	14.330	17.790	18.000
40%	4	0	0.064	22	10	14.726	19	9	14.586	24.118	24.000
50%	5	0	0.076	20	10	14.528	19	9	14.536	29.923	30.000
60%	3	0	0.046	21	10	14.610	20	9	14.426	35.977	36.000
70%	2	0	0.020	21	10	14.624	19	6	14.182	42.221	42.000
80%	1	0	0.008	22	10	14.594	19	7	13.158	47.889	48.000
90%	1	0	0.008	21	10	14.506	20	9	13.332	54.062	54.000
			Average	**21**	**10.0**	**14.616**	**19.4**	**7.9**	**13.882**		

Figure 10. Number of Node = 16.

	$\langle W_{ADD} \rangle$			$\langle W_{M1} \rangle$			$\langle W_{M2} \rangle$			# of Diff Conn Req. (from Simulation)	Expected # of Diff Conn Req.(Calculated)
	Max	Min	Avg	Max	Min	Avg	Max	Min	Avg		
10%	3	0	0.104	52	34	42.742	52	34	42.802	24.904	24.800
20%	3	0	0.114	52	33	42.988	54	32	42.716	49.400	49.600
30%	4	0	0.140	54	35	43.100	52	35	42.916	74.557	74.400
40%	2	0	0.074	52	34	43.020	52	34	42.802	98.931	99.200
50%	3	0	0.094	53	34	42.896	56	34	42.896	124.731	124.000
60%	4	0	0.086	52	34	42.714	52	36	42.634	148.447	148.800
70%	3	0	0.084	52	35	42.710	56	34	42.468	173.743	173.600
80%	3	0	0.046	53	34	42.834	53	34	42.614	198.260	198.400
90%	7	0	0.056	54	34	42.824	53	33	42.822	223.142	223.200
			Average	**53**	**34.1**	**42.870**	**53.3**	**34**	**42.741**		

Figure 11. Number of Node = 32.

6 Conclusion

In this paper, we addressed an issue on reconfiguring logical topologies in WDM optical rings. Specifically, we consider the problem of finding a sequence of lightpaths additions and deletions such that the logical topology remains connected in the presence of any single physical link failure throughout the reconfiguration.

We first discussed the complexity of the problem by exhibiting examples that require complicated designs of reconfigurations. We then presented a simple algorithm that can be implemented if a certain condition is satisfied. A limitation on implementing this simple algorithm is also discussed. Finally, we proposed a heuristic algorithm, that uses the minimum reconfiguration cost, with the objective of minimizing the total number of wavelengths used in reconfiguration.

Further work includes the development of algorithms that minimize the total reconfiguration cost when the total number of wavelengths is fixed.

Acknowledgement

This work was supported in part by the DARPA under grant N66001-00-18949 (co-funded by NSA), the DISA under LUCITE contract, and the NSF under grant ANI9973098.

References

[1] O. Crochat and J.-Y. Le Boudec, "Design Protection for WDM Optocal Networks," *IEEE J. Sel. Area Comm.*, 16(7): 1158-1165, Sept. 1998.

[2] H. Lee, H. Choi, S. Subramaniam, and H.-A. Choi, "Survivable Logical Topology Design in WDM Optical Ring Networks," *Proc. the 39th Annual Allerton Conference on Communications, Control, and Computing*, October, 2001, Invited Paper.

[3] E. Modiano, and A. Narula-Tam, "Survivable routing of logical topologies in WDM networks,", *Proc. Infocom*, pp.348-57, April 2001

ALGORITHM *MinCostReconfiguration*

Input: M_1, M_2, G_P
// M_1 is the survivable embedding of the current logical topology G_1 over physical topology G_P;//
// M_2 is a survivable embedding of a new logical topology to be reconfigured over G_P from G_1.//
// We assume that M_2 is obtained using the algorithm proposed in [2].//
Output: W_{add}
// Let W_{reconf} denote the total number of wavelengths used in reconfiguration using this algorithm;//
// Let W_{M_1} and W_{M_2} denote the number of wavelengths used in embeddings M_1 and M_2.//

1. Let $ADD = M_2 - M_1$ and $DELEETE = M_1 - M_2$.
// ADD is the set of lightpaths not in M_1 but in M_2 and//
// $DELETE$ is the set of lightpaths in M_1 but not in M_2.//
2. Let $W_{reconf} = \max\{W_{M_1}, W_{M_2}\}$.
3. **while** $ADD \neq \emptyset$ or $DELETE \neq \emptyset$ **do**
4. For any path $p \in ADD$, add a coresponding lightpath if the wavelength constraint is not violated, and repeat this process until no more addition is possible.
5. For any path $p \in DELETE$, delete p if the survivablity contraint is not violated, and repeat this process until no more deletion is possible.
6. Let $W_{reconf} \leftarrow W_{reconf} + 1$.
7. **endwhile**
8. Let $W_{reconf} \leftarrow W_{reconf} - 1$.
9. **Return** $W_{add} = W_{reconf} - \max\{W_{M_1}, W_{M_2}\}$.