Fast Restoration in WDM Mesh Networks*

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

Abstract

Past research on link protection in WDM mesh networks has focussed on achieving maximum possible restorability while trying to minimize the amount of necessary signaling after failures occur. Recognizing that different connections require different levels of protection and that it is not necessary to protect all links, we introduce a new perspective on the design of fast link protection in WDM mesh networks. Our experimental results show that it is possible to achieve signaling-free restoration for some network topologies studied in the literature with significantly reduced backup capacity. Our formulated problem is shown to be NP-complete, and an ILP formulation is given.

1 Introduction

As wavelength-routing paves the way for network throughputs of possibly hundreds of Tb/s, network survivability becomes critical. A short network outage can lead to data losses of the order of several gigabits. Hence, protection or dedicating spare resources in anticipation of faults, and rapid restoration of traffic upon detection of a fault are becoming increasingly important. According to [3], the overall availability requirements are of the order of 99.999% or higher. Survivability is the ability of the network to withstand equipment and link failures.

The main goals of survivable network design are to be able to perform rapid restoration (comparable to SONET's 50 ms) at as small a cost as possible (i.e., using minimum resources). Node equipment failures are typically handled using redundant equipment within the node (including redundant switches), and hence we focus on the failures of links, which occur due to backhoe accidents, and are usually dealt with by allocating redundant capacity on other network links and switching the affected traffic to the redundant capacity. Thus, a certain amount of redundant capacity called as protection capacity is pre-allocated for recovering from link failures.

In this paper, we concentrate on link protection or loopback protection wherein traffic over a link that fails is diverted to a backup path between the end-points of the link. Link protection is generally faster than path protection wherein the rerouting of traffic is done an end-to-end basis [7]. In order to save cost, the protection capacity can be shared

*This work was supported in part by the DARPA under grant N66001-00-18949 (co-funded by NSA), by the DISA under NSA-LUCITE contract, and by the NSF under grant ANI-9973098.

between links that are known to not fail simultaneously. However, capacity sharing also leads to increased switching times because the switches on the backup path must be configured after the failure happens.

Well-developed protection techniques are available for ring networks because of their popularity in SONET networks. Link protection is particularly attractive because of its fast restoration. Therefore, it is but natural to try similar techniques in mesh networks. These techniques generally try to embed cycles on a given mesh network and use these cycles as protection rings. However, if a link lies on more than one cycle, signaling after failures is required in general to resolve the ambiguity in the cycle to which the affected traffic must be rerouted.

In this paper, we propose a new scheme for link protection without signaling. The idea behind our approach can be summarized as follows. We first note that if the graph is an Eulerian graph (i.e., there exists a circuit that traverses every link exactly once), then every link has a backup path (by just following the Euler path connecting the two end-nodes of the failed link) and no signaling is necessary. This is a sufficient condition, but not necessary, because any path that connects two nodes on any cycle can also be protected by the cycle. For example, consider a non-Eulerian graph G shown in Figure 1 in which G has a cycle $C = [e_1, e_2, e_3, e_4, e_5, e_6, e_7]$, and two links $[e_8]$ and $[e_9]$ connecting the cycle. (We call such a link a *chordal link*, a link whose end-nodes belong to the same cycle.) A backup path $P(e_i)$ for each link e_i is then defined as $P(e_i) = C - \{e_i\}$ for $1 \leq i \leq 7$, $P(e_8) = [e_2, e_3, e_4]$, and $P(e_9) = [e_7, e_1]$.

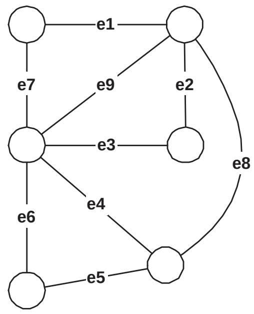


Figure 1: G

Observe that the hop-length of a backup path defined using the above scheme can be very long (e.g., the hop-length of $P(e_1)$ is 6). However, if not every link is required to be protected, the hop-length of each backup path may be reduced significantly (e.g., if link e_8 in Figure 1 is not protected, two cycles $[e_1, e_2, e_3, e_7]$ and $[e_4, e_5, e_6]$ can be used for defining a backup path of each link e_i for $1 \leq i \neq 8 \leq 10$, and the maximum hop-length then becomes only 3). As different connections may require different levels of protection (e.g., some connections may require paths composed of only protected links and others may not), an interesting problem formulation in certain network environments would then be to find a subset of links that need to be protected such that the network is fully connected (there exists a path between any pair of nodes) using only protected links. Desired objectives in such a formulation may include minimizing the average or maximum hop-length of protected paths and minimizing the backup capacity (i.e., the number of links that are used in one or more backup paths) of the networks, yet maximizing the link restorability.

A cycle-oriented preconfiguration of backup paths has been also considered in [5, 4], where the authors introduced *p-cycles*. A simple cycle (i.e., a cycle without having any node repeated in the cycle) is called a p-cycle, and a p-cycle is used to protect links in the cycle and chordal links. Two major differences exist between the method presented in [5, 4] and our approach. Firstly, while our approach only requires cycles to be link-disjoint (i.e., a node may appear more than once in any cycle), each p-cycle has to be simple. In Figure 1, for example, usin our approach, a cycle $C = [e_1, e_2, e_3, e_4, e_5, e_6, e_7]$ can be used as a protection cycle to protect all links in C as well as two other links e_8 and e_9. However, C cannot be defined as a p-cycle. Insteads, two simple cycles $[e_1, e_2, e_3, e_7]$ and $[e_4, e_5, e_6]$ may be identified as p-cycles, in which link e_8 cannot be protected using any of these cycles. An additional p-cycle is then required to provide a protection of link e_8 with extra redundant capacity. Secondly, their approach requires computing the set of all simple disctinct cycles up to some limiting size generated from the network topology to be used for selecting protection cycles. However, as it will be clear in the following section, we do not need to generate all possible cycles. Selecting cycles in our approach is based on the cencept of the Euler cycle, a well-known graph theoreic result.

In the following section, we formally give a formulation of our problem under a single link-failure model. Some numerical results are given in Section 3 in order to compare our approach with other existing approaches. The problem complexity is discussed in Section 4. An integer programming formulation for the problem is developed in Section 5. Section 6 concludes the paper.

2 Problem Formulation

In this section, we formally define a problem whose solution can be used to obtain signaling-free backup paths as we discussed earlier. Our problem formulation is general without specifying a specific optimization criterion, and various optimization properties may be added in this formulation to achieve a desired goal. It is assumed that the network topology is 2-edge connected since otherwise any cut-edge cannot be protected.

Link-Disjoint Cycles with Chordal Linkss (LDCCL): Given a graph $G = (V, L)$ with node set V and link set L, find a subset L' of L such that (i) $L' = A \cup B$, (ii) A is decomposed into link-disjoint (but not necessarily node-disjoint) cycles, (iii) B is decomposed into chordal links, and (iv) for any pair of nodes in V, there exists a path only using links in L'.

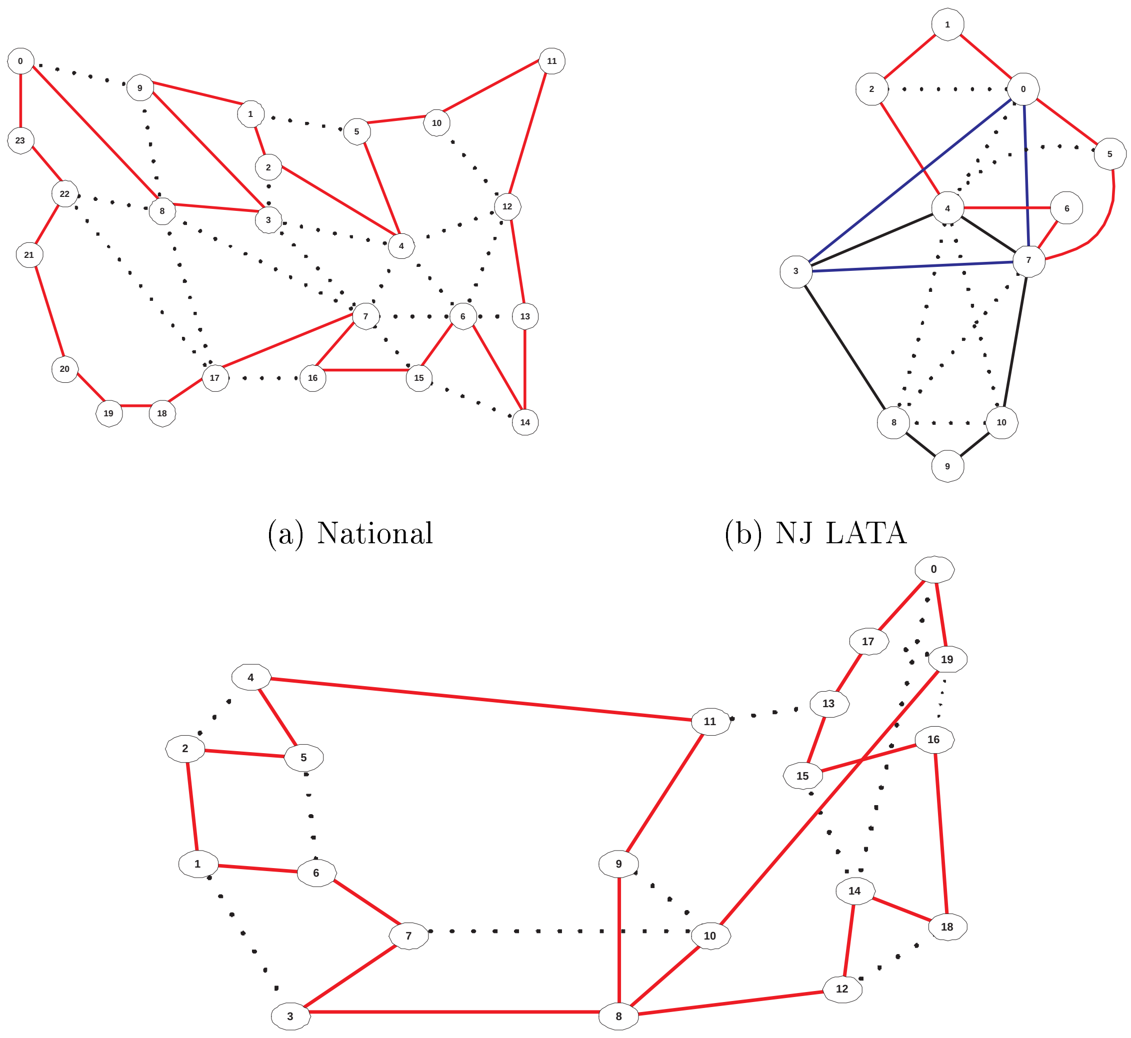
3 Numerical Results

We have obtained some numerical results in order to compare our approach with other existing approaches [8, 9]. Only single-link failures are considered. Both approaches in [8, 9] achieve 100% protection. Two different restorabilities are considered using our approach: (i) 100% and (ii) less than 100%.

We use the following performance metrics for comparison: number of protected links denoted by P; maximum and average backup path length denoted by H and H_{max} - the worst-case and average backup path length over all link failures; and number of links with 100% backup capacity denoted by B - the number of links that are used in one or more backup paths; and maximum and average number of cross-connects that need to be configured after a failure occurs denoted by S and S_{max} - the worst-case and average number of cross-connect switching over all backup paths.

We implemented our algorithm for LDCCL, the backup paths resulting from WDM loopback recovery method (WL) [9], and the backup paths resulting from double-cycle cover method (DCC) [8] in which the algorithm is guaranteed to provide a backup path for every link only when the graph is planar. As it turns out, two of the networks that we present results for are not planar, and we were only able to implement the algorithm suggested in [8] to find backup paths in one of those networks. Performance of the three algorithms are presented for 3 network topologies: National network, NJ LATA network, and ARPANET network - topologies that have been studied in the literature.

The backup paths used for 100% restorability are defined as follows. In Figure 2, links represented by solid lines form a single cycle in both (a) and (c). The backup path of each link is defined using these cycles. In (b), 3 cycles are defined as 0-1-2-4-6-7-5-0, 0-7-3-0, and 3-4-7-10-9-8-3. Dotted lines represent chordal links.



(a) National

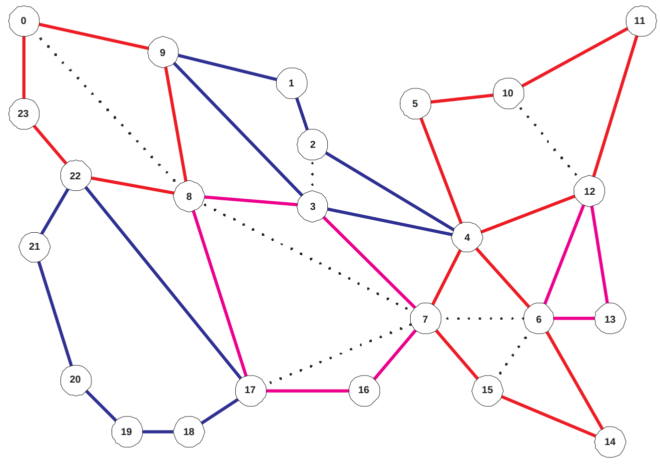
(b) NJ LATA

(c) ARPANET

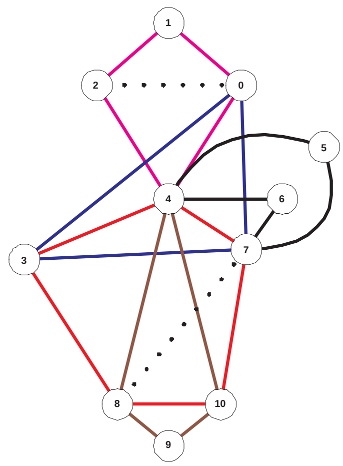
Figure 2: Network Topologies (DCCP with 100% Restorability)

The backup paths used for less than 100% restorability are defined as follows. Figure 3 shows only protected links in each topology. Note that each topology is still connected, i.e., there exists a path between any pair of nodes only using protected links. In (a), 7 cycles are defined as 0-9-8-22-23-0, 9-1-2-4-3-9, 8-3-7-16-17-8, 22-17-18-19-20-21-22, 5-

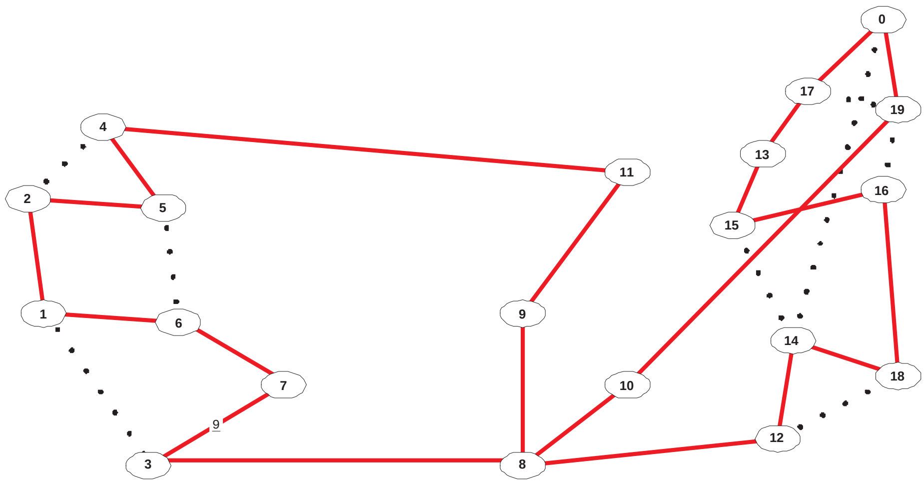
10-11-12-4-5, 12-13-6-12, and 4-6-14-15-7-4. In (b), 5 cycles are defined as 0-1-2-4-0, 4-5-7-6-4, 4-10-9-8-4, 0-7-3-0, and 4-7-10-8-3-4. In (c), two cycles are defines as 4-11-9-8-3-7-6-1-2-5-4 and 8-10-19-0-17-13-15-16-18-14-12-8. The dotted lines again represent chordal links.



(a) National



(b) NJ LATA



(c) ARPANET

Figure 3: Network Topologies (DCCP with less than 100% Restorability)

As shown in Table 1, the path lengths using our approach are relatively large if all the links must be protected. But if the number of links that need to be protected is reduced, then backup path lengths are considerably reduced. Furthermore, it was possible to achieve signaling-free restoration for all the 3 network topologies we considered using our approach. It should be finally observed that our approach has the potential for significantly reducing the backup capacity. The backup capacity of LDCCL with 100% restorability (when compared with the other two methods) was reduced by 45%, 35%, and 34%, respectively.

Algorithm	Metric	NATIONAL	NJ LATA	ARPANET
WL	P	44	23	32
	H	5.2	2.5	6.3
	H_{max}	10	4	12
	B	36	19	29
	S	1.86	1.09	3.53
	S_{max}	4	2	7
DCC	P	44	23	N/A
	H	6.2	2.7	N/A
	H_{max}	19	5	N/A
	B	44	23	N/A
	S	0	0	N/A
	S_{max}	0	0	N/A
LDCCL 100% Restorability	P	44	23	32
	H	13	4.1	14.25
	H_{max}	23	6	20
	B	24	16	21
	S	0	0	0
	S_{max}	0	0	0
LDCCL $< 100\%$ Restorability	P	41	22	29
	H	3.65	3.0	7.85
	H_{max}	5	4	10
	B	34	20	21
	S	0	0	0
	S_{max}	0	0	0

Table 1: Performance Comparisons

4 Problem Complexity

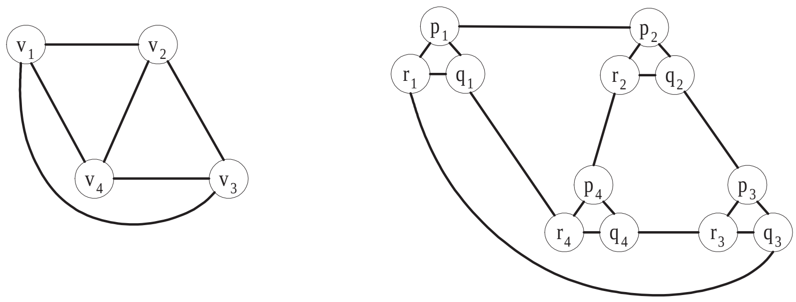
In this section, we consider the complexity of the LDCCL and show its NP-completeness. More precisely, the following result is presented. In our discussion, nodes and vertices are interchangeably used, and so are links and edges.

Theorem 4.1 *Given a 2-edge connected graph $G = (V, L)$, deciding whether there exists a subset L' of L satisfying conditions (i) - (iv) defined in the LDCCL is NP-complete.*

Proof: Our reduction is from the following known NP-complete problem: it is NP-complete [2] to decide whether a 3-regular graph G has a Hamiltonian cycle (HC), i.e., a cycle going through each node exactly once.

From a given 3-regular graph $G = (V, E)$ with vertex set $V(G) = \{v_i \mid 1 \le i \le n\}$, we construct a new graph G' as follows. The node set of G' is defined as $V(G') = \{p_i, q_i, r_i \mid 1 \le i \le n\}$ where each of p_i, q_i, and r_i corresponds to one of the edges in $E(G)$ incident at v_i. The link set of G' is defined as $L(G') = L_1 \cup L_2$, where $L_1 = \{(p_i, q_i), (q_i, r_i), (r_i, p_i) \mid 1 \le i \le n\}$ and $L_2 = \{(a_i, b_j) \mid a_i \in \{p_i, q_i, r_i\},\ b_j \in \{p_j, q_j, r_j\},$

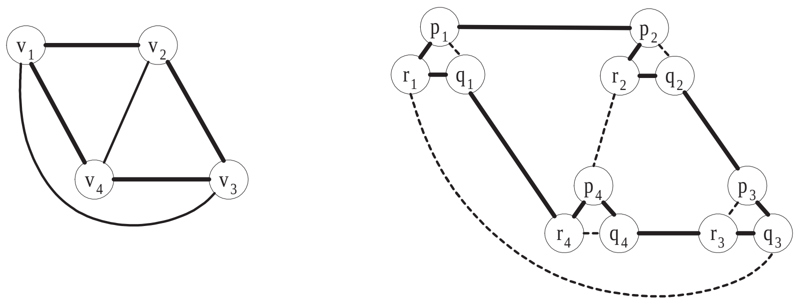
(a_i, b_j) corresponds edge $(v_i, v_j) \in E(G)\}$. Figure 4 shows an example of G and G' illustrating our construction scheme.



(a) G (b) G'

Figure 4:

From this construction, we claim that G has a HC if and only if there exists a subset of links $L' \subseteq L(G')$ satisfying conditions (i) - (iv). Suppose G has a HC, and let E_1 and E_2 be sets of edges in HC and not in HC, respectively. We then construct $L' \subset L(G')$ as follows. Let (v_i, v_j) and (v_j, v_k) be any two adjacent edges in E_1, where edge (v_i, v_j) corresponds to nodes p_i and q_j, and edge (v_j, v_k) corresponds to nodes r_j and p_k. We then include links (p_i, q_j), (r_j, p_k), (q_j, p_j) and (p_j, r_j) in L'. Similarly, other links are included in L' by considering every pair of adjacent edges in E_1. Figure 5 (a) shows a HC $v_1 - v_2 - v_3 - v_4 - v_1$ in G, and L' is shown in (b) where solid lines are links in L' and dotted lines are in $L - L'$.



(a) HC (b) L'

Figure 5:

Conversely, assume that G' has a desired subset of links $L' \subseteq L(G)$. We then note that links in L' form a single cycle going through every node in G' exactly once. This is based on the observation that each node in G' has to belong to a cycle and having more than one cycle cannot satisfy this condition since each node has only 3 adjacent edges. A HC in G can then be easily constructed from L', and this completes the proof. ■

5 ILP Formulation for LDCCL with Maximum Restorability

In this section, we add an optimization criterion, namely, maximizing the restorability, to the LDCCL and present its ILP formulation. We first introduce our notation and definitions, and then we describe a mathematical formulation of the problem and the given constraints.

5.1 Notation and Definitions

The LDCCL with maximum restorability is represented as $G(V, L)$ where V is the set of n network nodes and L is the set of m links in the network. For each link i, its two end-nodes are denoted as p_i and q_i. Each link with its two end-nodes p and q is denoted as i_{pq}. For each node p in V, $L(p)$ denotes the set of links adjacent at p, and $N(p)$ denotes the set of nodes adjacent to p.

For each link $i \in L$, two binary variables are defined: x_i and y_i which are equal to 1 if link i belongs to a cycle and a chordal link, respectively. For each pair of nodes p and q, we define the set of assignment variables $Z = \{z^k_{pq} \mid 1 \leq k \leq n-1\}$ as follows: z^k_{pq} is equal to 1 if there exists a path from node p to q using at most k links such that for each link i in the path, $x_i = 1$.

5.2 Optimization Problem

Our objective is to select $L' \subseteq L$ so as to maximize $|L'|$ satisfying conditions (i) - (iv) in the LDCCL formulation. This is defined to be the optimization problem identified as **P**:

$$\mathbf{P}: \quad \max \sum_{1 \leq i \leq m} (x_i + y_i) \tag{1}$$

subject to constraints on $\{x_i\}$, $\{y_i\}$, $\{z^k_{pq}\}$.

Constraints on variables $\{x_i\}$ are defined as follows.

$$(\sum_{i \in L(p)} x_i)/2 \; - \; \lfloor (\sum_{i \in L(p)} x_i)/2 \rfloor = 0 \qquad \forall p \in V \tag{2}$$

Constraint (2) ensures that each node has an even number of adjacent links that belong to cycles. (Note that the *flooring* function used in this equation is not linear, but implementation in an optimization solver is still feasible by setting an integer variable for the function.) This property can be used to find link-disjoint cycles (i.e., an Euler cycle from each component connected by links whose x_i values are 1.

In order to define variables $\{y_i\}$, we have to check if the two end-nodes p and q of link i are connected by links in the same cycle, i.e., $z^{n-1}_{pq} = 1$. Constraints on assignment variables $\{z^k_{pq}\}$ are defined as follows.

$$x_{i_{pq}}(1 - z^1_{pq}) + (1 - x_{i_{pq}}) z^1_{pq} = 0 \qquad \forall i_{pq} \in L \tag{3}$$

$$\sum_{q' \in N(q)} (1 - z_{pq}^k - z_{pq'}^{k-1} z_{q'q}^1 + 2 z_{pq'}^{k-1} z_{q'q}^1 z_{pq}^k) \geq 1 \quad \forall p, q \in N, \forall k \in \{2, \cdots, n-1\} \tag{4}$$

Constraint (3) ensures that $z_{pq}^1 = 1$ if and only if $x_{i_{pq}} = 1$. Constraint (4) ensures that $z_{pq}^k = 1$ if and only if there exists at least one node $q' \in N(q)$ such that $z_{pq'}^{k-1} = 1$ and $z_{q'q}^1 = 1$. Constraints on variables $\{y_i\}$ are now defined below.

$$y_i(1 - z_{i_p i_q}^{n-1}) + (1 - y_i) z_{i_p i_q}^{n-1} = 0 \qquad \forall i \in L \tag{5}$$

$$x_i + y_i \leq 1 \qquad \forall i \in L \tag{6}$$

Constraint (5) ensures that $y_i = 1$ if and only if $z_{i_p i_q}^{n-1} = 1$, i.e., two end-nodes i_p and i_1 of link link i are in the same cycle. Constraint (6) ensures that each link $i \in L$ can be either a cycle link or a chordal link, but it cannot be both.

The full connectivity constraint given in condition (iv) of the LDCCL is finally ensured by the following.

$$z_{pq}^{n-1} = 1 \qquad \forall p, q \in V \tag{7}$$

Note that if there exists a path only using protected links, i.e., each link i in the path has either $x_i = 1$ or $y_i = 1$, then there also exits a path connecting the same two end-nodes only using links whose x_i values are all 1. Hence, constraint (7) ensures the full-connectivity of the network using only protected links.

5.3 Remarks

Note that the LDCCL problem is NP-complete; hence, problem **P** is at least as hard as the LDCCL. This implies that there may not exist a solution to the LDCCL, and the result of the optimization may turn out to be 0.

In this section, we have considered the LDCCL with the objective of maximizing retorability. However, other optimization objectives such as minimizing the total backup capacity or the worst-case hop-length while maintaining all conditions (i) - (iv) in LDCCL are possible. The ILP formulation developed in this section clearly provides a framework for the LDCCL with other optimization criteria, and those variations will be discussed in a full paper.

6 Concluding Remarks

In this paper, we have introduced a new perspective on fast link protection in WDM mesh networks. The formulation of our problem considered in this paper was based on the recognition that different connections require different levels of protection and that it is not necessary to protect all links. The experimental results were developed for three well-known network topologies: National network, NJ Lata network, and ARPANET network. In all three topologies, 100% restorability with signaling-free link protection was possible with backup capacity significantly reduced compared with other existing methods. When the 100% restorability requirement was relaxed, our approach was shown to reduce other performance metrics such as worst-case and average hop-length of backup paths as well.

While our problem was shown to be NP-complete, we have developed a framework for an ILP formulation for our problem that can be used for the problem with other variations of optimization criteria.

Further research includes implementation of the developed ILP formulation in a commercial opmizer such as CPLEX and experiment for various other network topologies.

References

[1] G. Ellinas and T. Stern, "Automatic protection switching for link failures in optical networks with bidirectional links," Proc. GLOBECOM, pp. 152-156, Nov. 1996.

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

[3] O. Gerstel and R. Ramaswami, "Optical layer survivability: A service prospective," IEEE Communications Magazine, vol. 38, no. 3, pp. 104-113, March 2000.

[4] W. Grover and J. Doucette, "Design of a Meta-Mesh of Chain Subnetworks: Enhancing the Attractiveness of Mesh-Restorable WDM Networking on Low Connectivity Graphs," *IEEE J. on Selected Areas in Communications,* Vol. 20, No. 1, January 2002, pp. 47-61.

[5] W. Grover and D. Stamatelakis, "Cycle-Oriented Distributed Preconfiguration: Ring-like Speed with Mesh-like Capacity for Self-planning Network Restoration," *Proc. ICC* 1998, pp. 537-543.

[6] M Medard, S. G. Finn, and R. Barry, "WDM loop-back recovery in mesh networks," Proc. INFOCOM, pp. 752-759, March 1999.

[7] R. Ramaswami and K. N. Sivarajan, Optical Networks: A Practical Perspective, Morgan Kaufmann, 2002.

[8] G. Ellinas and T. Stern, "Automatic protection switching for link failures in optical networks with bidirectional links," Proc. GLOBECOM, pp. 152-156, Nov. 1996.

[9] M Medard, S. G. Finn, and R. Barry, "WDM loop-back recovery in mesh networks," Proc. INFOCOM, pp. 752-759, March 1999.