A Polynomial Time Algorithm for Computing Disjoint Lightpaths in Minimum Isolated Failure Immune WDM Optical Networks

Guoliang Xue, Ravi Gottapu, Xi Fang, Gabriel Silva, Dejun Yang, and Krishnaiyan Thulasiraman

***Abstract*—A fundamental problem in survivable routing in WDM optical networks is the computation of a pair of link-disjoint (or node-disjoint) lightpaths for a given pair of source and destination nodes. However, this problem is NP-hard in general mesh networks. As a result, heuristic algorithms and ILP formulations have been proposed. In this paper, we advocate to use 2-connected subgraphs of minimum isolated failure immune networks as the underlying topology for WDM optical networks. We present a polynomial time algorithm for computing a pair of link-disjoint lightpaths in such networks. Extension to the node-disjoint case is straightforward. Numerical results are presented to demonstrate the scalability and effectiveness of our algorithm.**

1. Introduction

A. Overview of Related Work

Optical networks implemented using wavelength division multiplexing techniques, called wavelength-routed networks, are considered promising candidates for the backbone high-speed wide area network [32**? ? ?** , 33]. Among other things, wavelength routing allows logical/virtual topologies to be built on top of physical topologies to reflect the traffic intensities between the various nodes as well as to provide reliable services by allowing for reconfiguration in the event of failures. Our focus is on survivable routing in all optical WDM networks where there is no wavelength converters. In such networks, data is transmitted along a *lightpath*, which is a path in the network with a chosen wavelength available on every link of the path [6]. Since optical links carry a very high volume of data, survivability is a very important issue. To establish a survivable connection between two nodes u and v, it is desirable to find a pair of link-disjoint (or node-disjoint) lightpaths connecting these two nodes. In this setting, one of the lightpaths is used as the working path, while the other is used as a backup. In the event of a single link (or node) failure, the connection can be quickly switched to the backup lightpath. If the network is not 2-edge connected, it is not always possible to find a pair of link-disjoint paths connecting a given pair of nodes. Therefore we are interested in networks whose underlying topology is 2-edge connected or 2-vertex connected (also called 2-connected). While survivability is an important issue, quality of service (QoS) is also often required. In the following we present a broad overview of the different issues in survivability and QoS provisioning in optical networks.

Xue, Fang, Silva, and Yang are with the School of Computing and Informatics at Arizona State University, Tempe, AZ 85287. Email: {xue, xi.fang, gsilva, dejun.yang}@asu.edu. Gottapu is with Amazon, Inc. Thulasiraman is with the School of Computer Science at the University of Oklahoma, Norman, OK 73019. Email: thulasi@ou.edu. This research was supported by NSF grant 0830739 and ARO grant W911NF-09-1-0467. The information reported here does not reflect the position or the policy of the federal government.

B. Disjoint Lightpath Routing

In a WDM optical network, each fiber link can carry a number of wavelengths $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_W\}$. A WDM network is commonly modeled by an undirected graph $G = (V, E, \Lambda)$, where V is the set of *vertices*, denoting *nodes* in the network; E is the set of *edges*, denoting *links* (or optical fibers) in the network; $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_W\}$ is the set of *wavelengths* and $\Lambda(e) \subseteq \Lambda$ is the set of wavelengths available on link e. The terms vertices and nodes are used interchangeably, as well as edges and links. In this model, each undirected link $[u, v]$ in the network represents a bidirectional link connecting u and v. Whenever a link is used by a connection, it is occupied in both directions.

In WDM networks, data packets are transmitted along *lightpaths* [6**?**]. A *lightpath* $P(s, t, \lambda)$ between nodes $s \in V$ and $t \in V$ on wavelength $\lambda \in \Lambda$ is an s–t path $\pi(s, t)$ in G which uses wavelength λ on every link of path $\pi(s, t)$. We assume that the nodes do not have wavelength converters (the use of wavelength converters is considered expensive in current WDM networks) and as a consequence, each lightpath must maintain the same wavelength throughout the entire path. This is known as the *wavelength continuity constraint*.

Cuts in fibers are considered to be one of the most common failures in optical networks, while failures of routers are also possible. Since each link has many wavelengths, several lightpaths may pass through the same link. Therefore the failure of a particular link e may affect many existing connections– all the connections whose lightpaths use link e. To tolerate a single link (node, respectively) failure in the network, the path protection scheme of fault management establishes an *active lightpath* and a link-disjoint (node-disjoint, respectively) *backup lightpath*, so that in the event of a link failure (node failure, respectively) on the active lightpath, data can be quickly re-routed through the backup lightpath. In the dedicated path protection scheme, an alternate path is maintained in a stand-by mode for every source-destination path used for data transmission. These paths are referred to as the *secondary* or *backup* path and the *primary* or *active* path, respectively.

Figure 1 shows a WDM network (with 8 nodes and 10 links) and 2 existing connections. The active and the backup paths between the nodes a and d are a-b-d on wavelength λ_1 and a-f-d on wavelength λ_2, respectively. The active and the backup paths between the nodes f and h are f-g-h on wavelength λ_2 and f-d-h on wavelength λ_1, respectively. Clearly, in order to tolerate any single link (node, respectively) failure, the backup path should not be sharing any fiber link (node, respectively) with its corresponding active path. Thus the backup path should be *link-disjoint* (node-disjoint, respectively) with the active path.

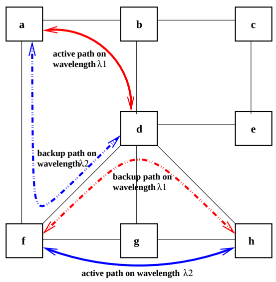


Fig. 1. Active and backup lightpaths in a WDM network are link-disjoint.

In order to compute a pair of link-disjoint active-backup lightpaths, the current literature uses what is known as the *shortest active path first* (APF) heuristic [11, 22], which first computes a shortest lightpath as a candidate for the active path, then finds the shortest lightpath that is link-disjoint from the candidate path. Computational studies show that the APF heuristic is quite effective in practice [11], but there is no performance guarantee for the APF heuristic. When the APF heuristic fails to find a pair of disjoint lightpaths, we would not know whether or not such a pair exists. Therefore researchers proposed integer linear programming (ILP) formulations [5, 25, 35**?** , 36]. Although the ILP formulation of the problem can be used to find optimal solutions, solving an ILP may take exponential time in the worst case.

In [3], Andersen *et al.* proved that the problem of computing a pair of disjoint (either link-disjoint or node-disjoint) lightpaths in a WDM network is NP-hard, asserting a commonly held belief in the WDM networking research community [4, 11]. They also designed an enhanced version of the commonly used active path first heuristic and presented simulation results showing that their new heuristic finds optimal solutions in 99.8% of the cases tested, while using only a fraction of the time required by the integer linear programming based algorithm. Yuan and Jue [46] independently proved the hardness of the disjoint lightpath routing problem almost at the same time. Hardness of a related problem [4] was proved by Hu [20]. Other related works can be found in [2, 10, 13, 16, 17, 23, 26, 27, 31, 34**?** **?** , 35].

C. Disjoint Paths in Networks

To make our review of literature as complete as possible we present below a review of works that relate to the problem of finding link/node disjoint paths in a network. Here the paths are not required to be lightpaths.

Given a pair of vertices, finding edge/vertex-disjoint paths (not lightpaths) between these vertices is a fundamental problem of great value in several applications such as fault tolerant routing in communication networks, network survivability and VLSI design. A more recent application is in the design of survivable logical topology mapping in IP over WDM Optical networks [1],[2]. In view of its importance, this problem has been studied extensively in the literature. Ford and Fulkerson proposed a polynomial-time algorithm for computing two paths with minimum total cost based on minimum-cost network flow model [3]. Suurballe and Tarjan presented algorithms that are more efficient [4], [5]. Xiao etal [6] ,[7] studied the problem of finding two or more disjoint paths that minimize total cost and also satisfy constraints on total delay of these paths. They provided approximation algorithms using integer linear programming formulations and Lagrangian relaxation. If in a given situation disjoint paths do not exist, then one may be interested in paths that satisfy certain constraints on the number of overlapping paths/nodes Recently, Wang *et al.* [8] provided algorithmic frameworks that can be used to solve a number of different variants of this problem.

Given 2k vertices s1,..., sk , t1,..., tk ,the problem of constructing k disjoint paths pi from si to ti (1 i k) is considerably more complicated than the case when only one pair of vertices is involved. Both, the edge- and the vertex-disjoint version of the problem, are NP-hard in the case of directed graphs. For a good review of current literature on this may be found in [9]. Reference [9] studied this problem extensively for the case of k =2. In particular, they showed that the $O(mn)$-time algorithm of Shiloach [10] can be modified to solve the 2-vertex-disjoint paths problem in only $O(n + m(m, n))$ time, where m is the number of edges in G, n is the number of directed graphs and denotes the inverse of the Ackermann function. They also improved the running time for the 2-edge-disjoint paths problem on undirected graphs as well as the running times for the 2-vertex- and the 2-edge-disjoint paths problem on dags.

Some of the other works related to the disjoint paths problem may be found in [11]-[16].

[1] K. Thulasiraman, M. S. Javed, and G. Xue, Circuits/Cutsets duality and a unified algorithmic framework for survivable logical topology design in IP-over-WDM optical networks, in INFOCOM, 2009, pp. 1026C1034. [2] K. Thulasiraman, M. Javed, and G. Xue, Primal meets dual: A generalized theory of logical topology survivability in IP-over-WDM optical networks, in Communication Systems and Networks (COMSNETS), 2010 Second International Conference on, 2010, pp. 1C10.

[3]] L. R. Ford and D. R. Fulkerson, Flows in Networks, Princeton Univ. Press, 1962.

[4] W. Suurballe, Disjoint paths in a network, Networks, vol. 4, pp. 125C145, 1974. [5] J.W. Suurballe and R. E. Tarjan, A quick method for 03nding shortest pairs of disjoint paths, Networks, vol. 14, pp. 325C336, 1984. [6]Ying Xiao, K. Thulasiraman, and Guoliang Xue, ”QoS Routing in Communication Networks: Approximation Algorithms Based on the Primal Simplex Method of Linear Programming”, IEEE Transactions on Computers, July 2006.

[7]Ying Xiao, K. Thulasiraman and Guoliang Xue, ”Constrained Shortest Link-Disjoint Paths Selection: A Network Programming Based Approach”, IEEE Transactions in Circuits and Systems, Vol 53, May 2006, pp 1174-1187.

***** [8]S. Q. Zheng, Jianping Wang, Bing Yang, and Mei Yang, , ”Minimum Cost Multiple Paths Subject to Minimum Link and Node Sharing in Networks” (with J. Wang, B. Yang, M. Yang), IEEE/ACM Transactions on Networking, Vol. 18, No. 5, pp. 1436-1449, 2010. *****

[9] Torsten Tholey, ”Solving the 2-Disjoint Paths Problem in Nearly Linear Time”, Theory of com;puting systems, Vol.3, 2006, 51-78.

[10] A. Itai, Y. Perl, and Y. Shiloach, The complexity of finding maximum disjoint paths with length constraints, Networks, vol. 12, pp. 277C286, 1982. [11] C. Li, S. T. McCormick, and D. Simchi-Levi, Finding disjoint paths with different path-costs: Complexity and algorithm, Networks, vol. 22, pp. 653C667, 1992. [12] C. Chekuri and S. Khanna, Edge-disjoint paths revisited, ACMTrans. Algor., vol. 3, no. 4, 2007, Article no. 46. [13] K. Ishida, Y. Kakuda, and T. Kikuno, A routing protocol for 03nding two node-disjoint paths in computer networks, in Proc. Int. Conf. Netw. Protocols, 1992, pp. 340C347. [14] R. Krishnan and J. Silvester, Choice of allocation granularity inmulti- path source routing schemes, in Proc. IEEE INFOCOM, 1993, vol. 1, pp. 322C329. [15] C. L. Li, S. T. McCormick, and D. Simchi-Levi, The complexity of 03nding two disjoint paths with min-max objective function, Discrete Appl. Math., vol. 26, no. 1, pp. 105C115, 1990.

[16] D. Xu, Y. Chen, Y. Xiong, C. Qiao, and X. He, On the complexity and algorithms for 03nding the shortest path with a disjoint counterpart, IEEE/ACM Trans. Netw., vol. 14, no. 1, pp. 147C158, Feb. 2006.

Throughout the paper, we will use the terms vertex (as well as edge and graph, respectively) and node (as well as link and network, respectively) interchangeably. We will concentrate on undirected networks. Therefore a link connecting nodes u and v is denoted by either $[u, v]$ or $[v, u]$.

2. Minimum Isolated Failure Immune Networks

In order to provide network survivability, a price to pay is redundancy in network elements. Here a network element can be either a link or a node. Complete graphs offer the most survivability, but have the highest redundancy–there is a link connecting every pair of nodes. Trees have no redundancy (among connected networks), but offer no survivability–both node failures and link failures are catastrophic for a tree network. A ring network can survive single node failure or single link failure, but two link failures will leave the network disconnected. Many methods have been proposed to design networks which are immune to certain types of node and link failures; most of them try to overcome failures by having a network with large connectivity [18]. In this section, we take a close view of the class of networks known as minimum isolated failure immune networks [12, 42].

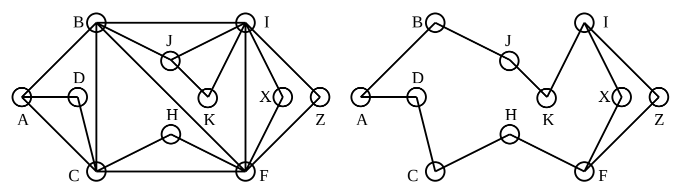
A. 2-Tree Networks

We start with the definition of 2-tree networks. We use $[u, v]$ to denote a link connecting nodes u and v. Since we deal with undirected networks, $[u, v]$ and $[v, u]$ both denote the same link. We use $\triangle xyz$ to denote the triangle with nodes x, y, z and links $[x, y]$, $[x, z]$ and $[y, z]$. The nodes in a triangle are not ordered. Therefore $\triangle xyz$, $\triangle yxz$, etc., all mean the same triangle.

Definition 2.1 (2-Tree Networks): A *2-tree* can be defined recursively as follows, and all 2-trees may be obtained in this way. A triangle is a 2-tree. Further, given a 2-tree and a link $[x, y]$ of the 2-tree, we can add a new node z adjacent to both x and y; the result is a 2-tree. □

Example 1: Fig.2(a) shows a 2-tree network. Initially, the triangle $\triangle ABC$ is a 2-tree with node set $\{A, B, C\}$ and link set $\{[A, B], [A, C], [B, C]\}$. Next, we add a new node D adjacent to both A and C ($[A, C]$ is a link in the current 2-tree), leading to the 2-tree with node set $\{A, B, C, D\}$ and link set $\{[A, B], [A, C], [B, C], [A, D], [C, D]\}$. To describe the above process, we say *node D expands out of link $[A, C]$*. We can continue this process to *expand* the 2-tree constructed so far to obtain the 2-tree shown in Fig.2(a).

One way to obtain the 2-tree shown in Fig.2(a) from the 2-tree on node set $\{A, B, C, D\}$ is to execute the following sequence of expansion operations. (1) Expand node F out of link $[B, C]$; (2) expand node H out of link $[C, F]$; (3) expand node I out of link $[B, F]$; (4) expand node J out of link $[B, I]$; (5) expand node K out of link $[I, J]$; (6) expand node X out of link $[I, F]$; (7) expand node Z out of link $[I, F]$. Another way to obtain the 2-tree shown in Fig.2(a) from the 2-tree on node set $\{A, B, C, D\}$ is to execute the following sequence of operations. (1) Expand node F out of link $[B, C]$; (2) expand node I out of link $[B, F]$; (3) expand node Z out of link $[I, F]$; (4) expand node X out of link $[I, F]$; (5) expand node J out of link $[B, I]$; (6) expand node H out of link $[C, F]$; (7) expand node K out of link $[I, J]$. □



(a) A 2-tree (b) A 2-connected partial 2-tree

Fig. 2. (a) A 2-tree on 11 nodes. This graph is isolated failure immune (IFI). (b) A 2-connected partial 2-tree. This graph is not isolated failure immune.

One can verify that every ring network is a subgraph of some 2-tree network that spans the same set of nodes. However, a 2-tree network might not have a ring subgraph that spans the same set of nodes. For example, the 2-tree in Fig.2(a) does not have a ring subgraph that spans all of its nodes. Fig.2(b) shows a minimum (defined by the number of links) 2-connected subgraph of the 2-tree in Fig.2(a).

Our algorithms rely on a *separation property* of 2-trees [39]: For every link $[x, y]$, the graph can be partitioned into one or more components, which pairwise intersect only in $[x, y]$, and the union of these components is the entire 2-tree. Moreover, each component so obtained is a 2-tree and the removal of x and y from a component does not disconnect the component. When this partition for link $[x, y]$ contains only a single component, the link is called a *peripheral*; otherwise, the link is called a *2-separator*. Furthermore, we have the following property ([39]).

Lemma 2.1 ([39]): Let $[x, y]$ be a 2-separator in a 2-tree $\mathbb{T}$. Let S_1 and S_2 be two of the components induced by the 2-separator $[x, y]$. Then

1) Let π_1 be a path within S_1 that does not use link $[x, y]$, and π_2 be a path within S_2 that does not use link $[x, y]$. Then π_1 and π_2 do have any common link, and do not have any common node other than x and y.
2) Let a be a node in S_1 other than x and y, and b be a node in S_2 other than x and y. Then any a–b path in $\mathbb{T}$ must go through either x or y. □

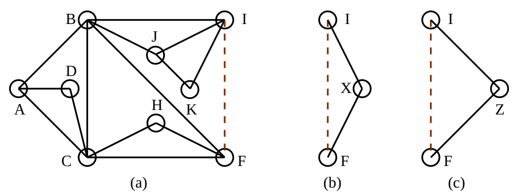


Fig. 3. The 2-tree in Fig.2(a) can be partitioned into 3 components which pairwise intersect at link $[I, F]$: (a) the component that contains node A; (b) the component that contains node X; (c) the component that contains node Z.

Example 2: We use the 2-tree in Fig.2(a) to illustrate the above concepts. The links $[A, C]$, $[B, C]$, $[B, F]$, $[B, I]$, $[C, F]$, $[F, I]$ and $[I, J]$ are 2-separators. All other links are peripherals.

The 2-tree can be partitioned into three components at link $[F, I]$, shown in Fig.3(a)-(c). The three components pairwise insert only at link $[F, I]$. Let π_1 be the path (F, C, B, I) and π_2 be the path (F, X, I). Clearly, the two paths do not share any common link, or any common node other than F and I. Since A and Z are in two different components, any path connecting A to Z must go through either node F or node I. □

B. Minimum Isolated Failure Immune Networks

Farley [12] studied an important class of networks, which can survive many simultaneous node and link failures, as long as they are *isolated*. Here two node failures are isolated if the nodes are not adjacent; two link failures are isolated if the links do not meet at a node; a node failure and a link failure are isolated if the failed link is not incident to the failed node.

Definition 2.2 (IFI Networks): A network is said to be *isolated failure immune (IFI)* if all pairs of functional nodes can still communicate, as long as the failures are pairwise isolated. An IFI network is called a *minimum IFI network*, if it has the minimum number of links among all IFI networks on the same number of nodes. □

Example 3: The network in Fig.2(a) is IFI. When nodes D, H, K, and links $[A, C]$, $[B, J]$ fail simultaneously (note that these network elements are pairwise isolated), the functional nodes are still connected. The simultaneous failures of links (F, Z) and (I, Z) will leave node Z disconnected from the other nodes. However, the links (F, Z) and (I, Z) are not isolated, because they are both incident to node Z. The simultaneous failures of nodes I and F will leave node X disconnected from the other functional nodes. Again, the two failures (nodes I and F) are not isolated.

The network in Fig.2(b) is 2-connected, but not IFI. For example, simultaneous failures of nodes B and C would leave nodes A and D disconnected from the other functional nodes. Note that nodes B and C are isolated. Also, simultaneous failures of link $[B, J]$ and node H (two isolated components) leaves node A disconnected from node Z. □

Farley [12] gave a detailed discussion of both the merits and the usage of IFI networks as survivable networks. In addition, he showed that IFI networks on n nodes have at least $2n - 3$ links. He further demonstrated that **every 2-tree is a minimum IFI network**. Later [42], Wald and Colbourn proved that **every minimum IFI network is a 2-tree**. They also studied a class of networks known as *partial 2-trees*: subgraphs of 2-trees. They presented a linear time algorithm to verify whether a given graph is a partial 2-tree, and when the graph is a partial 2-tree, to complete it to a 2-tree by adding artificial links. Both trees and rings are partial 2-trees. The nice structure of minimum IFI networks (and its subgraphs) have been exploited by many researchers in diverse applications. Many otherwise intractable problems can be solved efficiently on such networks [8, 9, 30, 42].

C. Survivable WDM Networks with Minimum IFI Topologies

A fundamental problem in survivable routing in WDM based all optical networks is the computation of a pair of link-disjoint (or node-disjoint) lightpaths connecting a source node with a destination node. Unfortunately, this problem is NP-hard [3, 20, 46]. Therefore ILP formulations and best-effort heuristic algorithms have been proposed to tackle this problem. However, ILP formulations are not scalable, and heuristic algorithms may fail to find a pair of link-disjoint lightpaths when one exists. Therefore a challenging fundamental problem in survivable optical network design is the following:

> *Is there a class of network topologies that can survive many failures and that a pair of node-disjoint (as well as link-disjoint) lightpaths in such networks can be computed efficiently?*

We believe that minimum IFI networks (or its 2-edge connected or 2-connected subgraphs) are an excellent candidate for the backbone of survivable optical networks. There are many reasons leading to our belief. First, the minimum IFI networks are fault-tolerant in the sense that they are immune to an arbitrarily number of isolated failures. Second, minimum IFI networks are sparse–for a network with n nodes, the number of links is only $2n-3$. This property enhances scalability. Third, minimum IFI networks have a nice structure that makes routing decisions much easier than in arbitrary networks [8, 9, 30, 42].

In this paper, we will show that the problem of computing a shortest pair of link-disjoint lightpaths is *polynomial time solvable* on minimum IFI networks (or their subgraphs). The node-disjoint counterpart can be solved in polynomial time using a similar (and simpler) algorithm. We choose to give a detailed presentation of the link-disjoint case in this paper.

3. Computing a Pair of Link-Disjoint Lightpaths in a Minimum IFI Network

We study link-disjoint lightpath routing in WDM networks whose underlying topology is a subgraph of a minimum IFI network. The problem is formally defined in the following.

Definition 3.1 (LDLP): Let $\mathbb{T} = (V, E)$ be a partial 2-tree with node set V and link set E. Let $\Lambda = \{1, 2, \dots, W\}$ be the set of wavelengths. For each link $e \in E$, the available wavelengths on link e is $\Lambda(e)$, which is a (possibly empty) subset of Λ. Let $s \in V$ be the source node, and $t \in V$ the destination node. The *link-disjoint lightpath routing problem* (**LDLP**) asks for a shortest pair of s–t link-disjoint lightpaths π_1 (on wavelength $\lambda_1 \in \Lambda$) and π_2 (on wavelength $\lambda_2 \in \Lambda$). In other words, we want to find a pair of s–t paths π_1 and π_2 such that

1) Paths π_1 and π_2 do not share a common link.
2) Wavelength λ_1 is available on every link on π_1.
3) Wavelength λ_2 is available on every link on π_2.
4) The path pair has the minimum total length (measured by the number of links) among all path pairs satisfying the above conditions. □

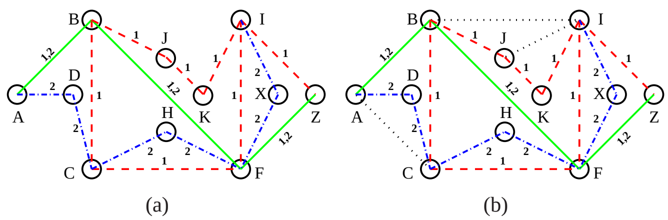


Fig. 4. (a) A WDM network ($n = 11, W = 2$) with a partial 2-tree topology. Link labels indicate the available wavelengths on each link. Links are also color/type coded: a red/dashed link has wavelength λ_1 available; a blue/dash-dot link has λ_2 available; a green/solid link has both λ_1 and λ_2 available.
(b) The same WDM network with a 2-tree topology. Links $[A, C]$, $[B, I]$, and $[I, J]$ are added by the algorithm of Wald and Colbourn to complete the partial 2-tree into a 2-tree. These *added* links have no wavelength available.

Fig.4(a) shows such a WDM network. For illustration purpose, we set $n = 11$ and $W = 2$. There are three kinds of links in the network: a dashed link (in red color) has only wavelength λ_1 available, and is indicated by the link label 1; a dash-dot link (in blue color) has only wavelength λ_2 available, and is indicated by the link label 2; a solid link (in green color) has both λ_1 and λ_2 available, and is indicated by the link label $1, 2$.

Example 4: Assume that A is the source and Z is the destination. The shortest A–Z lightpath on wavelength 1 (or 2) is $\pi_a = (A, B, F, Z)$. However, there does not exist another A–Z lightpath which does not share a common link with π_a. Clearly, $\pi_1 = (A, B, J, K, I, Z)$ on wavelength 1 and $\pi_2 = (A, D, C, H, F, Z)$ on wavelength 2 form a pair of link-disjoint lightpaths. We will see later that our algorithm can find a *shortest* pair of link-disjoint lightpaths: $\pi_1^{opt} = (A, B, F, I, Z)$ on wavelength 1 and $\pi_2^{opt} = (A, D, C, H, F, Z)$ on wavelength 2. *This example exposes the weakness of the Shortest Active Path First heuristic*: **Computing the shortest active path may fail to find a pair of disjoint lightpaths when such a pair exists**. *This example also demonstrates the advantage of our algorithm*: **When a pair of link-disjoint lightpaths exists, our algorithm guarantees finding a shortest pair.** □

To simplify the presentation, we define and study the **LDLP** problem with the wavelengths on π_1 and π_2 predetermined.

Definition 3.2 (LDLP-CW): Let $\mathbb{T}$, Λ, s, and t be the same as in Definition 3.1. Let $\lambda_1, \lambda_2 \in \Lambda$ be given. The *link-disjoint lightpath routing problem with chosen wavelengths* (**LDLP-CW**) asks for a shortest pair of s–t link-disjoint lightpaths π_1 on wavelength λ_1 and π_2 on wavelength λ_2. □

Our main results are the following: The MLDLP-CW problem can be solved in $O(n)$ time. The MLDLP problem can be solved in $O(nW^2)$ time, by solving $\frac{W(W+1)}{2}$ instances of the MLDLP-CW problem.

For any given WDM network with a partial 2-tree topology, one can apply the linear time algorithm of Wald and Colbourn to complete the partial 2-tree to a 2-tree. For example, the WDM network in Fig.4(a), although its underlying topology is a partial 2-tree that is not a 2-tree, can be assumed to be the WDM network in Fig.4(b) which as a 2-tree underlying topology. Note that any link added in the *completing process* will have no wavelength available–because the link simply does not exist in the physical network. However, assuming a 2-tree topology (rather than a partial 2-tree topology) will greatly simplify the description of our algorithm. Therefore, **without loss of generality, we will model the WDM optical network using a 2-tree** $\mathbb{T} = (V, E)$, where V is the set of n nodes, and E is a set of $2n - 3$ links.

The development of our algorithm for solving **LDLP** is organized in five subsections. In Section 3-A, we describe the attributes associated with each link. In Section 3-B, we describe the contraction operations. We also show the $O(1)$ time updates of the link attributes. In Section 3-C, we define the concept of *reduced 2-tree*, and the computation of a reduced 2-tree. In Section 3-D, we present our $O(n)$ time algorithm for **LDLP-CW**. Finally, in Section 3-E, we present our $O(nW^2)$ time algorithm for **LDLP**.

A. Link Attributes

Assume that wavelengths $\lambda_1, \lambda_2 \in \Lambda$ are chosen for the **LDLP-CW** problem. For each undirected link $[u, v]$, there are two *ordered node pairs* (u, v) and (v, u), one in each direction. For each ordered pair (u, v), we associate three attributes: $\alpha_1(u, v)$, $\alpha_2(u, v)$, and $\beta(u, v)$. The meaning and value of these attributes are explained in the following. We will use $G(u, v)$ to denote the subgraph currently represented by link (u, v). Before performing any contraction operations, $G(u, v)$ consists of the nodes u and v, and the link $[u, v] \in E$. When we perform a contraction operation, $G(u, v)$ will expand to include the subgraph that has been contracted to it. We will explain this in Section 3-B. We will use $l(\pi)$ to denote the length (measured by the number of links) of a lightpath π when π exists. $l(\pi)$ is set to ∞ when π is the empty path, denoted by ϕ.

- $\alpha_1(u, v)$ contains a shortest u–v lightpath $p_1(u, v)$ on wavelength λ_1 using only links in $G(u, v)$. If there is no u–v lightpath on λ_1 in $G(u, v)$, $\alpha_1(u, v)$ contains the empty path, denoted by ϕ.
- $\alpha_2(u, v)$ contains a shortest u–v lightpath $p_2(u, v)$ on wavelength λ_2 using only links in $G(u, v)$. If there is no u–v lightpath on λ_2 in $G(u, v)$, $\alpha_2(u, v)$ contains the empty path, denoted by ϕ.
- $\beta(u, v)$ is an ordered pair $(q_1(u, v), q_2(u, v))$ with shortest total length, where $q_1(u, v)$ contains an u–v lightpath on

λ_1 in $G(u,v)$, and $q_2(u,v)$ contains an u–v lightpath on λ_2 in $G(u,v)$ such that the two lightpaths are link-disjoint. The length of $\beta(u,v)$ is defined as the sum of the lengths of $q_1(u,v)$ and $q_2(u,v)$: $l(\beta(u,v) = l(q_1(u,v)) + l(q_2(u,v))$. If such a pair of link-disjoint lightpaths does not exist, we denote β by ϕ, and set its length to ∞. We also set both $q_1(u,v)$ and $q_2(u,v)$ to ϕ.

Therefore, for each undirected link $[u,v]$ in the graph, we have defined a total of six attributes: $\alpha_1(u,v)$, $\alpha_2(u,v)$, $\beta(u,v)$, $\alpha_1(v,u)$, $\alpha_2(v,u)$, $\beta(v,u)$. In general, $\alpha_1(u,v) = \phi$ if and only if $\alpha_1(v,u) = \phi$. When $\alpha_1(u,v) \neq \phi$, $p_1(v,u)$ contains the links in $p_1(u,v)$ in reverse order. Similarly, $\beta(u,v) = \phi$ if and only $\beta(v,u) = \phi$. When they are both non-empty, $q_1(v,u)$ contains the links in $q_1(u,v)$ in reverse order, and $q_2(v,u)$ contains the links in $q_2(u,v)$ in reverse order. *Given this relationship between $\beta(u,v)$ ($\alpha_1(u,v)$ and $\alpha_2(u,v)$, respectively) and $\beta(v,u)$ ($\alpha_1(v,u)$ and $\alpha_2(v,u)$, respectively), we will only show the value of one of them in our illustrations.*

Example 5: We will use the network in Fig.4(b) to illustrate the initial link attribute values. We assume $\lambda_1 = 1$ and $\lambda_2 = 2$ for our illustrations. Since for each link $[u,v]$ the current value of $G(u,v)$ is the link $[u,v]$ itself, there does not exist a pair of link-disjoint u–v lightpaths in $G(u,v)$. Therefore for each of the 19 links $[u,v]$ in Fig.4(b), we have $\beta(u,v) = \phi$.

Link $[A,C]$ has no wavelength available. Hence $\alpha_1(A,C) = \alpha_2(A,C) = \phi$. Same can be said about links $[B,I]$ and $[I,J]$.

Link $[A,B]$ has both λ_1 and λ_2 available. Hence $p_1(A,B)$ consists of the path (A,B), and $p_2(A,B)$ consists of the path (A,B) as well. The attribute values at links $[B,F]$ and $[F,Z]$ can be computed similarly.

Link $[B,C]$ has only λ_1 available. Hence $p_1(B,C)$ consists of the path (B,C), but $\alpha_2(B,C) = \phi$. The attribute values at links $[B,J]$, $[C,F]$, $[F,I]$, $[I,J]$, $[I,K]$, and $[I,Z]$ can be computed similarly.

Link $[A,D]$ has only λ_2 available. Hence $\alpha_1(A,D) = \phi$ and $p_2(A,D)$ consists of the path (A,D). The attribute values at links $[C,D]$, $[C,H]$, $[F,H]$, $[F,X]$, and $[I,X]$ can be computed similarly. □

B. Contraction Operations

We have seen in Section 2-A that a 2-tree is obtained by *expanding* a degree-2 node out of a link in an existing 2-tree. The reverse operation is the *contraction* of a degree-2 node z to the link $[x,y]$ with which it forms a triangle. The link attributes are updated according to certain rules when a contraction operation is performed.

We make use of the following two operators. The first operator is the concatenation operator $\circ$. If $p_i(x,z)$ and $p_i(z,y)$ are two lightpaths on wavelength λ_i, the concatenation of $p_i(x,z)$ and $p_i(z,y)$ is a lightpath from x to y on wavelength λ_i. We use $p_i(x,z) \circ p_i(z,y)$ to denote this concatenation, and the length is $l(p_i(x,z) \circ p_i(z,y)) = l(p_i(x,z)) + l(p_i(z,y))$. Clearly, $p_i(x,z) \circ p_i(z,y) = \phi$ if either $p_i(x,z) = \phi$ or $p_i(z,y) = \phi$. The second operator is the optimization operator opt. If $p_i^k(x,y)$ are x–y lightpaths on λ_i for $k = 1, 2, \ldots, K$, then $\mathsf{opt}\{p_i^1(x,y), \ldots, p_i^K(x,y)\}$ is denotes the shortest among these K paths. This notion also applies to pairs of lightpaths.

Lemma 3.1: Suppose that z is a degree-2 node, and $\triangle xyz$ is a triangle. Then we can contract node z as shown in Fig.5. The attributes associated with $[x,y]$ are updated according to the following rule.

$$p_1^{new}(x,y) = \mathsf{opt}\{p_1(x,y), p_1(x,z) \circ p_1(z,y)\}. \tag{3.1}$$

$$p_2^{new}(x,y) = \mathsf{opt}\{p_2(x,y), p_2(x,z) \circ p_2(z,y)\}. \tag{3.2}$$

$$\begin{aligned} & (q_1^{new}(x,y), q_2^{new}(x,y)) \\ = \quad & \mathsf{opt}\{(q_1(x,y), q_2(x,y)), \\ & (q_1(x,z) \circ q_1(z,y), q_2(x,z) \circ q_2(z,y)), \\ & (p_1(x,y), p_2(x,z) \circ p_2(z,y)), \\ & (p_1(x,z) \circ p_1(z,y), p_2(x,y))\}. \end{aligned} \tag{3.3}$$

$$p_1(x,y) = p_1^{new}(x,y); p_2(x,y) = p_2^{new}(x,y). \tag{3.4}$$

$$q_1(x,y) = q_1^{new}(x,y); q_2(x,y) = q_2^{new}(x,y). \tag{3.5}$$

PROOF. Fig. 5 illustrates the contraction operation. The shortest x–y lightpath on λ_1 in $G(x,y)$ after the contraction is either the shortest x–y lightpath on λ_1 in $G(x,y)$ before the contraction, or the concatenation of the shortest x–z lightpath on λ_1 in $G(x,z)$ and the shortest z–y lightpath on λ_1 in $G(z,y)$. Therefore we have equation (3.1). Equations (3.2) and (3.3) are similarly derived. For correct computations, we cannot overwrite the values of $p_1(x,y)$ and $p_2(x,y)$ before all the new values are computed. Therefore we introduced the notations p_i^{new} and q_i^{new}), and the equations (3.4) and (3.5). □

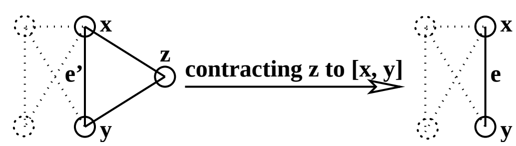
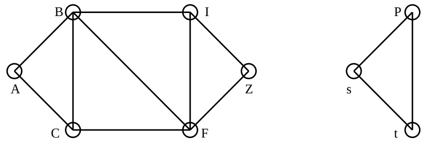


Fig. 5. If z is a degree-2 node which makes a triangle with link $[x,y]$, we can contract node z to link $[x,y]$. When we perform this contraction operation, the subgraph $G(x,y)$ is expanded to include also the union of $G(x,z)$ and $G(z,y)$. Therefore the subgraph represented by link e is the union of $G(e')$, $G(x,z)$, and $G(z,y)$. Also, the link attributes of e is computed from the link attributes of e', (x,z) and (z,y) in $O(1)$ time.

Example 6: In the 2-tree in Fig.4(b), node X is a degree-2 node which makes a triangle with link $[F,I]$. After contracting node X, the attributes at link $[F,I]$ are updated so that $p_1(F,I)$ remains to be the path (F,I), $p_2(F,I)$ is changed from ϕ to the path (F,X,I), and $\beta(F,I)$ is changed from ϕ to the path pair $((F,I),(F,X,I))$.

Since node K has degree-2, we can contract node K onto link $[I,J]$. After this operation, node J becomes degree-2. Therefore we can contract node J onto link $[B,J]$ At this time, $G(B,J)$ represents the subgraph induced by the nodes B, I, J, and K.

We can continue to contract node D and node H. After these contraction operations, we changed the graph in Fig.4(b) to the graph in Fig.6(a), where A and Z are the only degree-2 nodes.□



(a) (b)

Fig. 6. (a) The reduced 2-tree for Fig.4(b) with respect to A and Z. (b) A reduced 2-tree with respect to s and t if $[s,t]$ is a link in $\mathbb{T}$.

C. Reduced 2-Tree

The graph in Fig.6(a) has the following property: the only degree-2 nodes are the source node A and the destination node Z. We call such a graph a *reduced 2-tree*, which is formally defined in the following.

Definition 3.3 (Reduced 2-Tree): Let $\mathbb{T}$ be a 2-tree and s and t be two nodes in $\mathbb{T}$. We say that $\mathbb{T}$ is a *reduced 2-tree with respect to* s *and* t, if either $\mathbb{T}$ is a triangle or s and t are the only degree-2 nodes in $\mathbb{T}$. □

Lemma 3.2: Let $\mathbb{T}$ be a 2-tree, and s and t be two nodes in $\mathbb{T}$. Then $\mathbb{T}$ has a unique subgraph, which will be denoted by $\mathbb{T}(s,t)$, that is a reduced 2-tree with respect to s and t. Moreover, $\mathbb{T}(s,t)$ can be computed from $\mathbb{T}$ in $O(n)$ time by repeatedly contracting degree-2 nodes other than s and t. □

Lemma 3.2 can be proved using mathematical induction. We omit the proof, but point out that the existence and uniqueness of a reduced 2-tree with respect to two chosen nodes in a 2-tree network is analogous to the existence and uniqueness of a path connecting two nodes in a tree network.

Again let $\mathbb{T}$ denote the 2-tree in Fig.4(b). The network shown in Fig.6(a) is the reduced 2-tree $\mathbb{T}(A,Z)$. If we contract node A (the only degree-2 node other than the destination) in this reduced 2-tree, the result is another reduced 2-tree, which is $\mathbb{T}(C,Z)$. Now we can contract node C, which results $\mathbb{T}(B,Z)$. If we contract node B in the resulting 2-tree, we get the triangle $\triangle IFZ$. Recall that the sequence of nodes being contracted are A (which makes a triangle with 2-separator $[B,C]$), C (which makes a triangle with 2-separator $[B,F]$), B (which makes a triangle with 2-separator $[I,F]$). This sequence of 2-separators between the source node and the destination node in a reduced 2-tree is analogous to the sequence of nodes between the source node and the destination node in a path. Clearly, this sequence of 2-separators from the source node to the destination node is unique. We call $[B,C]$ the *predecessor* of $[B,F]$, and call $[B,F]$ the *successor* of $[B,F]$. A direct consequence of Lemma 2.1 is the following.

Lemma 3.3: Let $[P,Q]$ be any one of these 2-separators in $\mathbb{T}(s,t)$, and let p_1 and p_2 be two link-disjoint lightpaths connecting s to t such that p_1 is on wavelength λ_1 and p_2 is on wavelength λ_2. Then at least one of the following four statements is true.

1) Both p_1 and p_2 go through node P.
2) p_1 goes through node P, and p_2 goes through node Q.
3) p_1 goes through node Q, and p_2 goes through node P.
4) Both p_1 and p_2 go through node Q. □

D. A Linear Time Algorithm for LDLP-CW

Let $[P,Q]$ be a 2-separator of the reduced 2-tree $\mathbb{T}(s,t)$. Then it divides the reduced 2-tree into two parts, one containing the source node s, the other containing the destination node t. We will use $s(P,Q)$ to denote the part that contains the source node, with the link $[P,Q]$ excluded (as a technical convention). Assume that $s=A$ and $t=Z$. For the reduced 2-tree in Fig.6(a), $s(B,F)$ consists of the nodes A,B,C,F, and the links $[A,B]$, $[A,C]$, $[B,C]$, and $[C,F]$.

Let $[P,Q]$ be a 2-separator of $\mathbb{T}(s,t)$, or Q coincides with the destination node t. We associate with $[P,Q]$ the following four new attributes $\beta_1(P,Q)=\beta_4(Q,P)$, $\beta_2(P,Q)=\beta_3(Q,P)$, $\beta_3(P,Q)=\beta_2(Q,P)$, and $\beta_4(P,Q)=\beta_1(Q,P)$. Given the relationship $\beta_i(P,Q)=\beta_{5-i}(Q,P)$, we will concentrate on $\beta_i(P,Q)$ only.

For a given $i\in\{1,2,3,4\}$, $\beta_i(P,Q)$ is either empty (denoted by $\beta_i(P,Q)=\phi$) or is non-empty and has the following meaning.

- When nonempty, $\beta_1(P,Q)=\left(q_{11}^{PQ},q_{12}^{PQ}\right)$, where q_{11}^{PQ} is an s–P lightpath in $s(P,Q)$ on wavelength λ_1, q_{12}^{PQ} is an s–P lightpath in $s(P,Q)$ on wavelength λ_2, and the two lightpaths are link-disjoint. In addition, $l(q_{11}^{PQ})+l(q_{12}^{PQ})$ is minimized among all such lightpath pairs.
- When nonempty, $\beta_2(P,Q)=\left(q_{21}^{PQ},q_{22}^{PQ}\right)$, where q_{21}^{PQ} is an s–P lightpath in $s(P,Q)$ on wavelength λ_1, q_{22}^{PQ} is an s–Q lightpath in $s(P,Q)$ on wavelength λ_2, and the two lightpaths are link-disjoint. In addition, $l(q_{21}^{PQ})+l(q_{22}^{PQ})$ is minimized among all such lightpath pairs.
- When nonempty, $\beta_3(P,Q)=\left(q_{31}^{PQ},q_{32}^{PQ}\right)$, where q_{31}^{PQ} is an s–Q lightpath in $s(P,Q)$ on wavelength λ_1, q_{32}^{PQ} is an s–P lightpath in $s(P,Q)$ on wavelength λ_2, and the two lightpaths are link-disjoint. In addition, $l(q_{31}^{PQ})+l(q_{32}^{PQ})$ is minimized among all such lightpath pairs.
- When nonempty, $\beta_4(P,Q)=\left(q_{41}^{PQ},q_{42}^{PQ}\right)$, where q_{41}^{PQ} is an s–Q lightpath in $s(P,Q)$ on wavelength λ_1, q_{42}^{PQ} is an s–Q lightpath in $s(P,Q)$ on wavelength λ_2, and the two lightpaths are link-disjoint. In addition, $l(q_{41}^{PQ})+l(q_{42}^{PQ})$ is minimized among all such lightpath pairs.

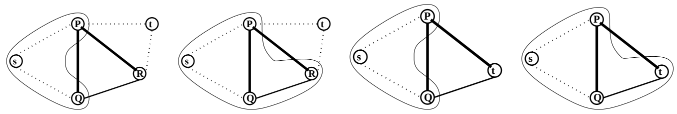
When $[P,Q]$ is the 2-separator that forms a triangle with the source node, we can compute the values of $\beta_i(P,Q)$ as follows.

Lemma 3.4: Let $[x,y]$ be the 2-separator that forms a triangle with the source node s in the reduced 2-tree, then the $\beta_i(x,y)$ can be computed as follows.

$$\beta_1(x,y) = \beta(s,x); \tag{3.6}$$
$$\beta_2(x,y) = (p_1(s,x),p_2(s,y)); \tag{3.7}$$
$$\beta_3(x,y) = (p_1(s,y),p_2(s,x)); \tag{3.8}$$
$$\beta_4(x,y) = \beta(s,y), \tag{3.9}$$

where $p_i(\bullet,\bullet)$ and $\beta(\bullet,\bullet)$ are defined in Section 3-A and computed according the rules in Section 3-B. □
The proof is straightforward, and is omitted.

Having computed the $\beta_i(P,Q)$ values for 2-separator $[P,Q]$, we can compute the $\beta_i(P,R)$ values for its succeeding 2-separator $[P,R]$ (if $[P,Q]$ is the 2-separator which makes a triangle with the destination node, we can compute the values $\beta_i(P,R)$). This is illustrated in Fig.7.



(a) (b) (c) (d)

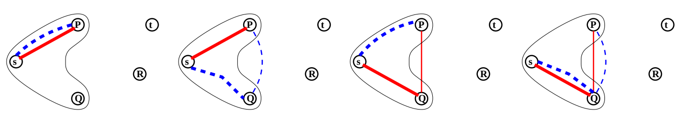
Fig. 7. The values of $\beta_i(P,R)$ can be computed from the values of $\beta_i(P,Q)$ in $O(1)$ time. (a) and (b): 2-separator $[P,R]$ is the successor of 2-separator $[P,Q]$. (c) and (d): 2-separator $[P,Q]$ makes a triangle with the destination node t, which coincides with R.

The rules for computing $\beta_i(P,R)$ from the values of $\beta_i(P,Q)$ are given in the next four lemmas. Their proofs are straightforward (illustrated in Figs.8-10, and are omitted.

Lemma 3.5: Let $[P,Q]$ be a 2-separator in the reduced 2-tree with respect to source s and destination t. Let $[P,R]$ be the successor of $[P,Q]$, as shown in Fig.7(a), or $[P,R]$ is a peripheral with R coincides with t, as shown in Fig.7(c). Then $\beta_1(Q,R)=(q_{11}^{PR},q_{12}^{PR})$ is the shortest path pair among the following four (possibly empty) path pairs:

case-1.1: $(q_{11}^{PQ},q_{12}^{PQ})$.
case-1.2: $(q_{21}^{PQ},q_{22}^{PQ}\circ p_2(Q,P))$.
case-1.3: $(q_{31}^{PQ}\circ p_1(Q,P),q_{32}^{PQ})$.
case-1.4: $(q_{41}^{PQ}\circ q_1(Q,P),q_{42}^{PQ}\circ q_2(Q,P))$.

$\beta_1(P,Q)=\phi$ when all of the above four path pairs are ϕ. □



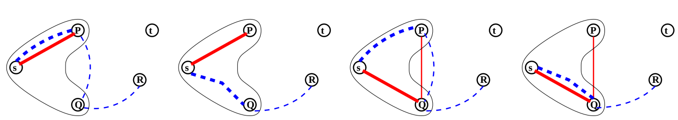
(a) caze-1.1 (b) case-1.2 (c) case-1.3 (d) case-1.4

Fig. 8. Computing the attributes $\beta_1(P,R)$. Selecting the best among four possible cases. (a) Using $\beta_1(P,Q)$; (b) Using $\beta_2(P,Q)$; (c) Using $\beta_3(P,Q)$; (d) Using $\beta_4(P,Q)$.

Lemma 3.6: Let $[P,Q]$ be a 2-separator in the reduced 2-tree with respect to source s and destination t. Let $[P,R]$ be the successor of $[P,Q]$, as shown in Fig.7(a), or $[P,R]$ is a peripheral with R coincides with t, as shown in Fig.7(c). Then $\beta_2(Q,R)=(q_{21}^{PR},q_{22}^{PR})$ is the shortest path pair among the following four (possibly empty) path pairs:

case-2.1: $(q_{11}^{PQ},q_{12}^{PQ}\circ p_2(P,Q)\circ p_2(Q,R))$.
case-2.2: $(q_{21}^{PQ},q_{22}^{PQ}\circ p_2(Q,R))$.
case-2.3: $(q_{31}^{PQ}\circ q_1(Q,P),q_{32}^{PQ}\circ q_2(P,Q)\circ p_2(Q,R))$.
case-2.4: $(q_{41}^{PQ}\circ p_1(Q,P),q_{42}^{PQ}\circ p_2(Q,R))$.

$\beta_2(P,Q)=\phi$ when all of the above four path pairs are ϕ. □



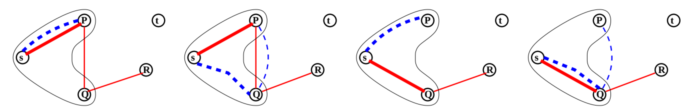
(a) case-2.1 (b) case-2.2 (c) case-2.3 (d) case-2.4

Fig. 9. Computing the attributes $\beta_2(P,R)$. Selecting the best among four possible cases. (a) Using $\beta_1(P,Q)$; (b) Using $\beta_2(P,Q)$; (c) Using $\beta_3(P,Q)$; (d) Using $\beta_4(P,Q)$.

Lemma 3.7: Let $[P,Q]$ be a 2-separator in the reduced 2-tree with respect to source s and destination t. Let $[P,R]$ be the successor of $[P,Q]$, as shown in Fig.7(a), or $[P,R]$ is a peripheral with R coincides with t, as shown in Fig.7(c). Then $\beta_3(Q,R)=(q_{31}^{PR},q_{32}^{PR})$ is the shortest path pair among the following four (possibly empty) path pairs:

case-3.1: $(q_{11}^{PQ}\circ p_1(P,Q)\circ p_1(Q,R),q_{12}^{PQ})$.
case-3.2: $(q_{21}^{PQ}\circ q_1(P,Q)\circ p_1(Q,R),q_{22}^{PQ}\circ q_2(Q,P))$.
case-3.3: $(q_{31}^{PQ}\circ p_1(Q,R),q_{32}^{PQ})$.
case-3.4: $(q_{41}^{PQ}\circ p_1(Q,R),q_{42}^{PQ}\circ p_2(Q,P))$.

$\beta_3(P,Q)=\phi$ when all of the above four path pairs are ϕ. □



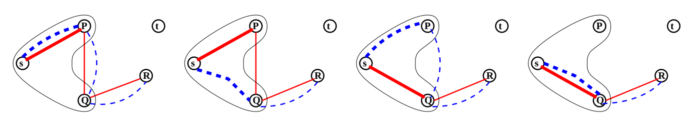
(a) case-3.1 (b) case-3.2 (c) case-3.3 (d) case-3.4

Fig. 10. Computing the attributes $\beta_3(P,R)$. Selecting the best among four possible cases. (a) Using $\beta_1(P,Q)$; (b) Using $\beta_2(P,Q)$; (c) Using $\beta_3(P,Q)$; (d) Using $\beta_4(P,Q)$.

Lemma 3.8: Let $[P,Q]$ be a 2-separator in the reduced 2-tree with respect to source s and destination t. Let $[P,R]$ be the successor of $[P,Q]$, as shown in Fig.7(a), or $[P,R]$ is a peripheral with R coincides with t, as shown in Fig.7(c). Then $\beta_4(Q,R)=(q_{41}^{PR},q_{42}^{PR})$ is the shortest path pair among the following four (possibly empty) path pairs:

case-4.1: $(q_{11}^{PQ}\circ q_1(P,Q)\circ q_1(Q,R),q_{12}^{PQ}\circ q_2(P,Q)\circ q_2(Q,R))$.
case-4.2: $(q_{21}^{PQ}\circ p_1(P,Q)\circ q_1(Q,R),q_{22}^{PQ}\circ q_2(Q,R))$.
case-4.3: $(q_{31}^{PQ}\circ q_1(Q,R),q_{32}^{PQ}\circ p_2(P,Q)\circ q_2(Q,R))$.
case-4.4: $(q_{41}^{PQ}\circ q_1(Q,R),q_{42}^{PQ}\circ q_2(Q,R))$.

$\beta_4(P,Q)=\phi$ when all of the above four path pairs are ϕ. □



(a) case-4.1 (b) case-4.2 (c) case-4.3 (d) case-4.4

Fig. 11. Computing the attributes $\beta_4(P,R)$. Selecting the best among four possible cases. (a) Using $\beta_1(P,Q)$; (b) Using $\beta_2(P,Q)$; (c) Using $\beta_3(P,Q)$; (d) Using $\beta_4(P,Q)$.

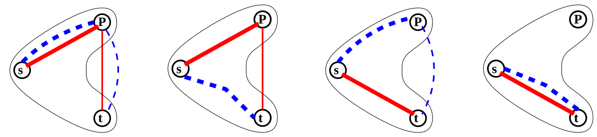
Now we have most of the building blocks for our algorithm. Let us use the reduced 2-tree in Fig.6(a) as an example. Using Lemma 3.4, we can compute the values for $\beta_i(B,C)$ for $i=1,2,3,4$. Then we can compute the values for $\beta_i(B,F)$, then for $\beta_i(I,F)$, then for $\beta_i(I,Z)$. The solution to **LDLP-CW** can be decided by the following lemma.

Lemma 3.9: Let $[P,Q]$ be a 2-separator in the reduced 2-tree with respect to source s and destination t. Assume that the destination node t makes a triangle with link $[P,Q]$, as shown in Fig.7(d). Then a solution to **LDLP-CW** is the path pair (π_1,π_2), which is the shortest path pair among the following four (possibly empty) path pairs:

case-F.1: $(q_{11}^{Pt}\circ q_1(P,t),q_{12}^{Pt}\circ q_2(P,t))$.
case-F.2: $(q_{21}^{Pt}\circ p_1(P,t),q_{22}^{Pt})$.
case-F.3: $(q_{31}^{Pt},q_{32}^{Pt}\circ p_2(P,t))$.
case-F.4: $(q_{41}^{Pt},q_{42}^{PQ})$.

The instance of **LDLP-CW** has no solution when all of the above four path pairs are ϕ. □

We summarize our design of the algorithm in Algorithm 1.



(a) case-F.1 (b) case-F.2 (c) case-F.3 (d) case-F.4

Fig. 12. Selecting the best among four possible cases. (a) Using $\beta_1(P,t)$; (b) Using $\beta_2(P,t)$; (c) Using $\beta_3(P,t)$; (d) Using $\beta_4(P,t)$.

Algorithm 1: Alg-LDLPCW

Input: A WDM optical network $\mathbb{T}$ with a 2-tree topology. A source node s. A destination node t. Two chosen wavelengths $\lambda_1 \in \Lambda$ and $\lambda_2 \in \Lambda$.

Output: A pair of link-disjoint s–t lightpaths p_1 on λ_1 and p_2 on λ_2 if such a pair exists.

1: **for** each link (u,v) in $\mathbb{T}$ **do**
2: Initialize $\alpha_1(u,v)$, $\alpha_2(u,v)$, $\beta(u,v)$, $\alpha_1(v,u)$, $\alpha_2(v,u)$, $\beta(v,u)$, according to the rules in Section 3-A.
3: **end for**
4: **while** there is a degree-2 node z other than s and t **do**
5: Assume z makes a triangle with link $[x,y]$. Contract node z and update $\alpha_1(x,y)$, $\alpha_2(x,y)$ and $\beta(x,y)$ according to the rules in Lemma 3.1.
6: **end while**
$\{$By now we have computed the the reduced 2-tree.$\}$
7: Let $\triangle xys$ be the triangle in the reduced 2-tree that contains s. Initialize $\beta_1(x,y)$, $\beta_2(x,y)$, $\beta_3(x,y)$, and $\beta_4(x,y)$ according to Lemma 3.4.
8: **while** there is a degree-2 node v other than t **do**
9: Let $[P,Q]$ be the 2-separator that forms a triangle with v. Let $[P,R]$ be the successor of $[P,Q]$, or $[P,R]$ be a peripheral and R coincides with t.
10: Compute $\beta_i(P,R)$ for $i=1,2,3,4$ according to Lemmas 3.5-3.8.
11: Delete node v.
12: STOP if $\beta_1(P,R)=\beta_2(P,R)=\beta_3(P,R)=\beta_4(P,R)=\phi$.
13: **end while**
14: Compute the shortest pair of link-disjoint lightpaths (π_1,π_2) according to Lemma 3.8. STOP if such a pair does not exist.
15: Extract the link-disjoint paths π_i and π_2 using a top-down method.

Example 7: I need to check the steps in this example. We use the network in Fig.4(b) to illustrate the algorithm. By the time the reduced 2-tree in Fig.6(a) is constructed, we have to following non-empty link attributes (for each link $[u,v]$, we only list the attribute for one ordered pair): $p_1(A,B) = p_2(A,B) = (A,B)$; $p_2(A,C) = (A,D,C)$; $p_1(B,I) = (B,J,K,I)$; $q_1(C,F) = p_1(C,F) = (C,F)$; $q_2(C,F) = p_2(C,F) = (C,H,F)$; $p_1(B,F) = p_2(B,F) = (B,F)$; $q_1(I,F) = p_1(I,F) = (I,F)$; $q_2(I,F) = p_2(I,F) = (I,X,F)$; $p_1(I,Z) = (I,Z)$; $p_1(F,Z) = p_2(F,Z) = (F,Z)$.

The sequence of 2-separators of the reduced 2-tree are $[B,C]$, $[B,F]$, and $[I,F]$.

At $[B,C]$, we find $\beta_1(B,C) = \phi$, $\beta_2(B,C) = ((A,B),(A,D,C))$, $\beta_3(B,C) = \phi$, and $\beta_4(B,C) = ((A,B,C),(A,D,C))$.

At $[B,F]$, we find $\beta_1(B,F) = \phi$, $\beta_2(B,F) = ((A,B),(A,D,C,H,F))$, $\beta_3(B,F) = \phi$, and $\beta_4(B,F) = ((A,B,C,F),(A,D,C,H,F))$.

At $[I,F]$, we find $\beta_1(I,F) = \phi$, $\beta_2(I,F) = ((A,B,J,K,I),(A,D,C,H,F))$, $\beta_3(I,F) = \phi$, $\beta_4(I,F) = ((A,D,C,H,F),(A,B,C,F))$.

At $[I,Z]$, we find $\beta_1(I,Z) = ((A,B,J,K,I),(A,D,C,H,F,X,I))$, $\beta_2(I,Z) = ((A,B,J,K,I),(A,D,C,H,F,Z))$, $\beta_3(I,Z) = ((A,B,F,Z),(A,D,C,H,F,X,I))$, $\beta_4(I,Z) = \phi$.

Finally, in line 13 of the algorithm, we compute the shortest pair of A–Z lightpaths are $((A,B,J,K,I,Z),(A,D,C,H,F,Z))$. While in the illustrations, we have listed out the paths during each step. In actual implementations, we only need to remember the length of the path, as well as its composition (the concatenation of one or two paths which are previously computed). Once we reached the final answer, the pair of paths can be extracted out in $O(n)$ time. □

Our analysis through this section leads to the following theorem.

Theorem 3.1: The worst-case time complexity of Algorithm 1 is $O(n)$. If there exists a pair of link-disjoint s–t paths π_1 and π_2 such that λ_1 is available on each link on π_1 and that λ_2 is available on each link on π_2, Algorithm 1 correctly finds such a pair of paths with minimum total length. Otherwise, Algorithm 1 stops without outputting any path. □

PROOF. Each contraction operation operation takes $O(1)$ time. There are $O(n)$ contraction operations. This leads to the time complexity of the algorithm. The correctness of the algorithm follows from Lemma 3.1 and Lemmas 3.3-3.8. □

E. Linear Time Algorithm for LDLP

Now we have developed an algorithm for solving **LDLP-CW**, we can use it as a subroutine to design an algorithm for **LDLP**. The basic idea is to solve **LDLP-CW** for all possible pairs of (λ_1,λ_1), and choose the shortest path pair computed as the solution. The algorithm is listed as Algorithm 2.

Algorithm 2: Alg-LDLP

Input: A WDM optical network $\mathbb{T}$ with a 2-tree topology. A source node s. A destination node t.

Output: A pair of link-disjoint lightpaths p_1 (on λ_1) and p_2 (on λ_2).

1: **for** $\lambda_1 = 1,2,\ldots,W$ **do**
2: **for** $\lambda_2 = \lambda_1,\lambda_1+1,\ldots,W$ **do**
3: Apply Alg-MLDLPCW to compute a pair of link-disjoint lightpaths
4: Keep the shortest pair computed so far
5: **end for**
6: **end for**

Theorem 3.2: Algorithm 2 correctly solve the **LDLP** problem in $O(nW^2)$ time. □

4. Numerical Results

To demonstrate the effectiveness of our algorithm, we have implemented our algorithm and tested it randomly generated partial 2-tree topologies. We show two sets of results here. First, we compare our optimal algorithm (denoted by **LDLP**) with the well-known shortest active path first heuristic (denoted by **SAPF**). Then we study the time complexity of our algorithm.

To compare **LDLP** with **SAPF**, we use the same network state information and the same connection request, given by a source-destination pair. Then we use both algorithms to find a pair of link-disjoint paths. There are four possible results. (1) Neither algorithm finds a pair of link-disjoint paths. In this case, the connection request is rejected. (2) **LDLP** finds such a pair, but **SAPF** fails. In this case, we make the link reservations in the network according to the paths found by **LDLP**, and update the network state information. (3) Both algorithms find a pair of paths, but the path found by **LDLP** is shorter. In this case, we make the link reservations in the network according to the paths found by **LDLP**, and update the network state information. (4) Both algorithms find a pair of paths, and the two pairs are of equal length. In this case, we make the link reservations in the network according to the paths found by **LDLP**, and update the network state information.

We repeat the above for K iterations, and consider the number of times each of these four cases happens. This result is shown in Fig.**??**.

To study the scalability of our algorithm, we study the time complexity of our algorithm as a function of n and W. This result is shown in Fig.**??**. We can see that the running time is proportional to n for any fixed W, and is proportional to W^2 for any fixed n.

5. Conclusions

In this paper, we have shown that computing a pair of link-disjoint lightpaths can be done in polynomial time, provided that the underlying topology is a subgraph of a minimum isolated immune network. Although not presented here, the result also holds for the node-disjoint case, where the corresponding is slightly simpler. Given the importance of isolated failure immune networks and the hardness of the disjoint lightpath routing problem in geneal mesh networks, we expect our result to have significant impact on the design of future survivable networks.

References

[1] S. Ahuja, S. Ramasubramanian, and M. Krunz; SRLG failure localization in all-optical networks using monitoring cycles and paths; *IEEE Infocom'2008*, accepted.

[2] V. Anand and C. Qiao, Dynamic establishment of protection paths in WDM networks, *IEEE ICCCN'2000*, pp. 198-204.

[3] R. Andersen, F. Chung, A. Sen, G. Xue; On disjoint path pairs with wavelength continuity constraint in WDM networks; *IEEE Infocom'2004*, pp. 524-535.

[4] R. Bhandari, *Survivable networks: algorithms for diverse routing*, Kluwer Academic Publishers, 1999.

[5] D. Banerjee and B. Mukherjee; Wavelength-routed optical networks: linear formulation, resourcebudgeting tradeoffs, and a reconfiguration study; *IEEE/ACM Transactions on Networking*, Vol. 8 (2002), pp. 598-607.

[6] I. Chlamtac, A. Ganz, and G. Karmi, Lightpath communications: an approach to high bandwidth optical WAN's, *IEEE Transactions on Communications*, Vol. 40(1992), pp. 1171-1182.

[7] H. Choi, S. Subramaniam, and H.A. Choi, On double-link failure recovery in WDM optical networks, *IEEE Infocom'2002*, pp. 808-816.

[8] C.J. Colbourn and G. Xue; A linear time algorithm for computing the most reliable source on a series-parallel graph with unreliable edges; *Theoretical Computer Science*; Vol. 209(1998), pp. 331-345.

[9] C.J. Colbourn and G. Xue; Grade of service Steiner trees in series-parallel networks; in Ding-Zhu Du, J.M. Smith, and J.H. Rubinstein, eds., *Advances in Steiner Trees*; Kluwer Academic Publishers, 2000, pp. 163-174.

[10] O. Crochat and J. L. Boudec, Design protection for WDM optical Networks, *IEEE JSAC*, Vol. 16(1998), pp. 1158-1165.

[11] D.A. Dunn, W.D. Grover and M.H. MacGregor, Comparison of k-shortest paths and maximum flow routing for network facility restoration, *IEEE JSAC*, Vol. 2(1994), pp. 88-99.

[12] A.M. Farley; Networks immune to isolated failures; *Networks*, Vol. 11 (1981), pp. 255-268.

[13] A. Fumagalli, I. Cerutti, M. Tacca, F. Masetti, R. Jagannathan and S. Alagar, Survivable networks based on optimal routing and WDM self healing rings, *IEEE Infocom'1999*, pp. 726-733.

[14] A. Fumagalli, L. Valcarenghi, IP restoration versus WDM protection: Is there an optimal choice?, *IEEE Networks*, Vol. 14, No. 6(2000), pp. 34-41.

[15] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman Publishers, San Francisco, 1978.

[16] N. Ghani, S. Dixit and T. S. Wang, On IP-over-WDM integration", *IEEE Communications Magazine*, Vol. 38, No. 3, pp. 72–84, March 2000.

[17] K.P. Gummadi, M.J. Pradeep and C.S.R. Murthy, An efficient primary-semenged backup scheme for dependable real-time communication networks, *IEEE/ACM Transactions on Networking*, Vol. 11(2003), pp. 81–94.

[18] S.L. Hakimi and A.T. Amin; On the design of reliable networks; *Networks*, Vol. 3(1973), pp. 241-260.

[19] P.H. Ho and H.T. Mouftah; A framework of service guaranteed shared protection for optical networks; *IEEE Communications Magazine*, Vol. 40, No. 2(2002), pp. 97-103.

[20] J. Q. Hu, Diverse routing in optical mesh networks, *IEEE Transactions on Communications*, Vol. 51(2003), pp. 489-494.

[21] M. Javed, K. Thulasiraman and G. Xue; Lightpaths routing for single link failure survivability in IP-over-WDM networks; *KICS/IEEE Journal of Communication and Networks*; Vol. 9(2007), pp. 394–401.

[22] A. Jukan, *AoS-based Wavelength Routing in Multi-Service WDM Networks*, Springer-Verlag/Wien, 2001.

[23] S. Kim; X.J. Zhang and S.S. Lumetta; Rapid and efficient protection for all-optical WDM mesh networks; *IEEE JSAC*, Vol. 25(2007), No. 9, pp. 68-82.

[24] M. Kodialam and T.V. Lakshman, Dynamic routing of locally restorable bandwidth guaranteed tunnels using aggregated link usage information, *IEEE Infocom'2001*, pp. 376-385.

[25] R.M. Krishnaswamy and K. N. Sivarajan, Design of logical topologies: a linear formulation for wavelength-routed optical networks with no wavelength changers, *IEEE/ACM Transactions on Networking*, Vol. 9(2001), pp. 186–198.

[26] C. Liu and L. Ruan; A new survivable mapping problem in IP-over-WDM networks *IEEE JSAC*, Vol. 25(2007), No. 4, pp. 25-34.

[27] Y. Liu and D. Tipper, Successive survivable routing for node failures, *IEEE Globecom'2001*, pp. 2093-2097.

[28] S.S. Lumetta and M. Medard, Towards a deeper understanding of link restoration algorithms for mesh networks, *IEEE Infocom'2001*, pp. 367-375.

[29] S.S. Lumetta, M. Medard, and Y. Tseng, Capacity versus robustness: A tradeoff for link restoration in mesh networks, *Journal of Light Wave Technology*, Vol. 18(2000), pp. 1765-1775.

[30] K. Miura, D. Takahashi, S.-I. Nakano, and T. Nishizeki; A linear time algorithm to find four independent spanning trees in four connected planar graphs, *International Journal of Foundations of Computer Science*, Vol. 10(1999), pp. 195–210.

[31] G. Mohan, C.S. RamMurthy, and A.K. Somani; Efficient algorithms for routing dependable connections in WDM optical networks; *IEEE/ACM Transactions on Networking*, Vol. 9 (2001), pp. 553-566.

[32] B. Mukherjee, *Optical Communication Networks*, McGraw-Hill, 1997.

[33] B. Mukherjee, WDM optical networks: progress and challenges, *IEEE JSAC*, Vol. 18(2000), pp. 1810–1824.

[34] C. Qiao and D. Xu; Distributed partial information management (DPIM) schemes for survivable networks - Part I, *IEEE Infocom'2002*, pp. 302-311.

[35] S. Ramamurthy and B. Mukherjee; Survivable WDM mesh networks. Part I-Protection; *IEEE Infocom'1999*, pp. 744-751.

[36] S. Ramamurthy and B. Mukherjee, Survivable WDM mesh networks. II. Restoration; *IEEE ICC'1999*, pp. 2023-2030.

[37] S. Ramasubramanian; Supporting multiple protection strategies in optical networks; *IEEE/ACM Transactions on Networking*, accepted.

[38] S. Ramasubramanian and A. Chandak; Dual-link failure resiliency through backup link mutual exclusion; *IEEE/ACM Transactions on Networking*, accepted.

[39] D.J. Rose, On simple characterizations of k-trees, *Discrete Mathematics*, Vol. 7 (1974), 317-322.

[40] L. Sahasrabuddhe, S. Ramamurthy and B. Mukherjee; Fault management in IP-over-WDM networks: WDM protection versus IP restoration; *IEEE JSAC*, Vol. 20(2002), pp. 21–33.

[41] J.W. Suurballe; Disjoint paths in a network, *Networks*; Vol. 4(1974), pp. 125-145.

[42] J.A. Wald and C.J. Colbourn; Steiner trees, partial 2-trees, and minimum IFI networks; *Networks*; Vol. 13 (1983), 159-167.

[43] H. Wang, E. Modiano and M. Medard, Partial path protection for WDM networks: End-to-end recovery using local failure information, *IEEE ISCC'02*, pp. 719-725.

[44] K. Xi, J. Chao; IP fast rerouting for single-link/node failure recovery; *IEEE BROADNETS'2007*.

[45]

[46] S. Yuan and J.P. Jue; Dynamic lightpath protection in WDM mesh networks under wavelength-continuity and risk-disjoint constraints; *Computer Networks Journal (Elsevier)*, Vol. 48(2005), pp. 91-112.

[47] J. Zhang, K. Zhu and B. Mukherjee; Backup reprovisioning to remedy the effect of multiple link failures in WDM mesh networks; *IEEE JSAC*, Vol. 24(2006), No. 8, pp. 57-67.