Circuits/Cutsets Duality and a Unified Algorithmic Framework for Survivable Logical Topology Design in IP-over-WDM Optical Networks

Krishnaiyan Thulasiraman and Muhammad S. Javed
School of Computer Science
University of Oklahoma
Norman, OK, USA
{thulasi, javed }@ou.edu

Guoliang (Larry) Xue
Department of Computer Science and Engineering
Arizona State University
Tempe, AZ, USA
xue@asu.edu

Abstract: **Given a logical topology G_L and a physical topology G, the survivable logical topology design problem in an IP-over-WDM optical network is to map the logical links into lightpaths in G such that G_L remains connected after the failure of any edge in G. In view of its fundamental nature and its practical importance, this problem has received considerable attention in the literature. The SMART algorithmic framework based on the circuits in $G_L$is a novel and very significant contribution to this problem. Taking advatange of the dual relationship between circuits and cutsets in a graph, we first present in this paper the primal algorithm CIRCUIT-SMART (similar to SMART) and algorithm CUTSET-SMART that is dual of CIRCUIT-SMART and proofs of correctness of these algorithms. To guarantee survivability we add additional logical links called protection edges, if necessary. This investigation has provided much insight into the structural properties of solutions to this problem and the structure of survivable logical graphs. Specifically, we present a highly simplified version of CUTSET-SMART that always provides a survivable mapping as long as G is 3-edge connected, and a survivable logicl topology structure. We also present algorithm INCIDENCE-SMART that uses incidence sets that are special cases of a cut. Two efficient heuristics, one based on maximum matching theory and the other based on both the primal and dual algorithms are also presented. Simulation results comparing the different algorithms in terms of computational time, protection capacity and survivability success rate are also presented.**

Keywords: Circuits, cutsets and duality; IP-over-WDM; lightpath routing; survivable networks; disjoint paths.

I. INTRODUCTION

IP-over-WDM networks are realized by *embedding* an IP network into a physical WDM network using IP routers and optical cross-connects (OXCs) [1]. It is common to refer to the IP network as the logical topology and the WDM network as the physical topology. Similarly, IP routers are referred to as logical nodes and the OXCs as WDM nodes. The logical nodes are connected to each other via logical connections called logical links (or edges) and physical nodes are connected through optical fibers called physical links. It is also common to assume that a logical node is always a node in the physical topology and all the physical nodes may not be present in the logical topology.

A logical topology is embedded into a physical topology by finding all-optical paths between the source and the destination of all the IP links in the physical topology and allocating wavelengths to the paths. Such paths, known as *lightpaths*, once established, may pass through multiple physical links but do not require any buffering or processing at the intermediate nodes [2]. The embedding of a logical link as a lightpath in the physical topology is also called a *mapping* of the link. The set of all such mappings defines the mapping of a logical topology into the physical topology [3].

In an IP-over-WDM network, a single fiber can carry several lightpaths, each using a several gigabits per second (Gbps) bit-rate wavelength. Usually, several such optical fibers are bundled together to form a cable to achieve terabits per second (Tbps) networks [4]. The cables are then laid along other utility services such as telephone lines, water pipes, cable etc. that require regular maintenance and upgrades, which leads to frequent fiber cuts due to human or natural events [4]. A cable cut disconnects all the lightpaths passing through the cable, resulting in tremendous loss of data that could deteriorate the performance of the entire network significantly [4]. This fact has drawn considerable attention to designing IP-over-WDM networks that continue to provide an acceptable level of service in the presence of fiber or network equipment failures. Such networks are generally called *survivable networks* or more precisely, *node survivable* or *link survivable networks* based on the type of failure [4].

A network can be hedged against node failures by providing redundant equipment that can be used in case of a node failure. On the other hand, identifying the precise location of fiber failure and performing the repairs may result in significant down time [4]. Therefore, a significant amount of literature is dedicated to designing link survivable IP-over-WDM networks, which is also the focus of this paper.

Protection and *restoration* are the two most widely discussed mechanisms for providing survivability in IP-over-WDM networks [5]. Protection is generally provided at the physical layer during the design stage. First *primary* or *working lightpaths* are found for the logical links that normally carry the

The work of K. Thulasiraman has been supported in part by NSF grants ANI-0312435 and ECS 04-26831 and the work of G. Xue has been supported by NSF grants 0830739 and 0721803.

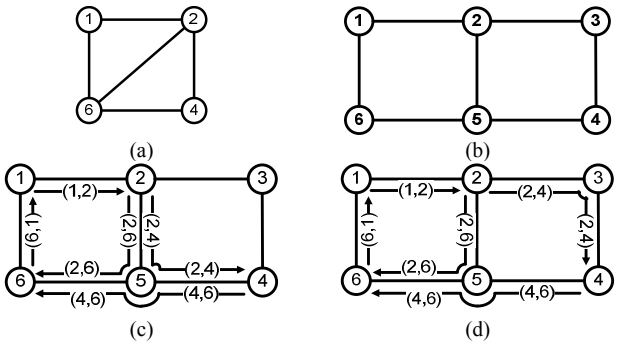


Figure 1. Illustration of mapping and survivability for general networks. (a) A logical topology (b) A physical topology (c) An unsurvivable mapping (d) A survivable mapping.

traffic and then *backup* or *protection lightpaths* are calculated that do not use physical links already assigned to the primary lightpaths (i.e. their mappings are disjoint) [5]. When a failure occurs, the network traffic carried by an affected primary lightpath is always switched to its corresponding backup lightpath. Since protection requires explicit reservation of resources, it is generally considered inefficient but very fast [5].

Restoration is generally more resource efficient but slow. Restoration is generally provided at the logical layer by provisioning the network with some additional capacity, which can be utilized by the IP routers to find backup lightpaths for the working lightpaths after a failure [5]. However, IP routers can find backup paths only if the logical links are mapped onto physical topology in such a way that the logical topology remains connected after the failure [4].

This requires that the logical and physical topologies are at least two-edge connected (i.e. they remain connected after the removal of a link), and finding link disjoint mappings for all or an appropriate subset of logical links. Such a mapping of the logical topology in the physical topology that does not cause the logical topology to be disconnected after a single physical link failure is called a *link survivable mapping*.

Fig. 1(a) and Fig. 1(b) show a logical topology and a physical topology, respectively. Fig. 1(c) shows an unsurvivable mapping of this logical topology. In this case, not all the mappings are disjoint and the logical topology is not survivable. For example, the failure of physical link (4,5) disconnects the logical topology. Fig. 1(d) shows a survivable mapping. In this case also, it can be seen that not all the mappings are disjoint and a physical link failure may disconnect multiple logical links but the logical topology still remains connected. For example, if the physical link (5,6) fails, logical links (2,6) and (4,6) get disconnected but it is possible to reach all the logical nodes through the remaining logical links. It can be observed that finding disjoint mappings for only a subset $\{(1,2),(2,4),(4,6),(6,1)\}$ is sufficient to guarantee survivability in this example.

The above example illustrates the important role played by the pair-wise (mutually) disjoint paths problem in finding survivable mappings. The problem of finding pair-wise disjoints paths is well studied and is NP-complete in general [6]. However, it is possible to find pair-wise disjoint paths in some special cases e.g. when the topology is undirected three edge-connected and the number of pair-wise disjoint paths required is two [6].

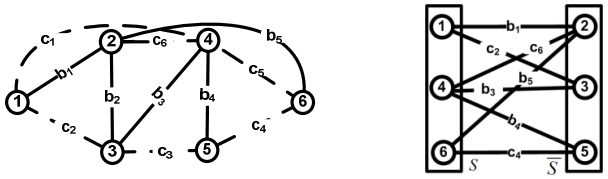
In [7,8], Modiano and Narula-Tam formally show that the problem of finding survivable mappings is NP-complete for general as well as for ring logical topologies. Therefore, they provide integer linear programs (ILPs) to find a solution. The ILP is based on the observation that a logical topology can get disconnected after the failure of a physical link only if the physical link carries an entire cut of the logical topology, or alternatively, every cut of the logical topology must contain at least a pair of edges with pair-wise disjoint mappings in order for the mappings to be survivable. However, the ILP does not scale well as it must examine all the possible cuts, a number that grow exponentially with the size of the topology [7, 8].

In [9] Todimala and Ramamurthy, based on [7,8], provide an improved ILP that applies to Shared Risk Link Groups (SRLG). The ILP incorporates wavelength assignment constraints and only considers primary cuts, but does not scale well either. However, when applied to planar cycles and hierarchical planar cycles, the ILP can be solved fairly quickly. In [10] Ducatelle and Gambardella also utilize the results from [7,8] and rather than evaluating all the cutsets, employ a probability function as an estimate of the cutsets.

Crochat et al. provide in [3] a comprehensive framework for designing survivable IP-over-WDM networks and define three constraints that must be respected by a solution. They note that the problem is NP-complete and suggest a heuristic based on Tabu search. Shenai and Sivalingam suggest in [11] a hybrid approach to survivability that uses a combination of restoration and protection.

In [5,12,13] Kurant and Thiran provide a framework called SMART (Survivable Mapping Algorithm by Ring Trimming). SMART utilizes circuits to find survivable mappings for logical topologies. The framework repeatedly picks connected *pieces* (subgraphs) of the logical topology and finds survivable mappings for these pieces. If a survivable mapping is found for a piece, its links are short-circuited (contracted) and the algorithm proceeds by picking another piece. The process is repeated until the logical topology is reduced to a single node or a search for a piece with survivable mapping is unsuccessful. If the logical topology is reduced to a single node, a survivable mapping for the logical topology has been found; otherwise a survivable mapping does not exist.

Motivation for the research reported in this paper is as follows. Duality between circuits and cuts in a graph is one of the well studied topics in graph theory. This concept has played a significant role in the development of methodologies for solving problems in various applications. Most of the early results in electrical circuit theory were founded on the duality relationship between circuits and cuts [14]. There is a wealth of literature on the role of duality in network optimization (that is, discrete optimization on graphs and networks) [15]. Most often, for a primal algorithm based on circuits there is a dual algorithm based on cuts for the same problem. The primal and dual algorithms possess certain characteristics that make one superior to the other depending on the application. SMART algorithm for the survivable logical topology design problem is based on circuits [5]. The question then arises whether there



(a) (b)

Figure 2. (a) A graph with a spanning tree (bold lines) (b) A cut.

exists a dual methodology based on cuts. In this paper we establish such an algorithm and variants of this algorithm that are computationally very efficient. Our work also provides much insight into the structure of solutions for the survivable topology design problem.

The rest of the paper is organized as follows. In section II, we review several results that highlight the dual relationship between circuits and cuts and the dual graph operations of deletion and contraction of an edge. We also introduce certain new concepts and results that will be the foundation of the methodologies developed in subsequent sections. In section III algorithm CIRCUIT-SMART that is essentially the SMART algorithm using fundamental circuits is presented. To guarantee survivability we add additional logical links called protection edges, if necessary. A new and simple proof of correctness using the dual results of section II is also provided. In section IV algorithm CUTSET-SMART which is the dual of CIRCUIT-SMART and a proof of its correctness are presented. We also present in this section a simplified version of CUTSET-SMART that requires mapping only certain pairs of logical edges into mutually disjoint lightpaths in the physical topology, in contrast to CUTSET-SMART and CIRCUIT-SMART that may require mapping more than two logical edges into mutually disjoint lightpaths. Using an earlier result we show that this simplified version will always result in a survivable mapping as long as the physical topology is 3-edge connected. We also present a survivable structure that has a powerful role to play. It can be used to augment the original logical graph, whenever needed, to guarantee survivability. In section V a 2–phase heuristic that combines CIRCUIT-SMART and CUTSET-SMART, and a maximum matching based heuristic are presented. In section VI algorithm INCIDENCE-SMART that uses incidence sets is presented (Incidence sets are special cases of cuts). Simulation results comparing the different algorithms of earlier sections are presented in section VII. We conclude in section VIII.

II. CIRCUITS AND CUTSETS DUALITY

Duality between circuits and cuts in a graph has been extensively studied and plays a fundamental role in several applications [14,15]. Deleting an edge and contracting an edge are also dual operations. In this section, we present several concepts and results relating to this duality. These results provide the basis for the algorithmic frameworks presented in the following sections. For standard graph theoretic concepts not defined here [14] may be consulted.

Consider a connected undirected graph $G(V, E)$ with vertex set V and edge set E. Without loss of generality, we assume that there are no parallel edges or self loops in G. Let G have $|V| = n$ vertices (or nodes) and $|E| = m$ edges (or links).

A connected acyclic subgraph of G containing all the n nodes is called a **spanning tree** $\boldsymbol{T}$ of G. The edges of a spanning tree T are called **branches** of T. The remaining edges of G are called chords with respect to T. We may also refer to chords as non-tree edges.

Consider a partition $(S, \bar{S})$ of vertex set V. Here $\bar{S}$ denotes the complement of S in V, i.e. $\bar{S} = V - S$. Then the set of edges with one node in S and the other in $\bar{S}$ is called a **cut** of G.

For example, consider the graph G in Fig. 2(a). Here the vertices are numbered $1,2, \dots ,6$. The bold edges in this figure denote the branches of a spanning tree T of G and the dotted edges are the chords of this tree. The partition $(S, \bar{S})$ with $S = \{1,4,6\}$ and $\bar{S} = \{2,3,5\}$ defines the cut shown in Fig. 2(b).

Adding a chord c to a spanning tree T produces exactly one circuit. This is called the **fundamental circuit** (in short, *f*-circuit) of T with respect to the chord c. We denote this circuit as $B(c)$. For example, if we add chord c_1 to the tree in Fig. 2(a) we get the fundamental circuit $B(c_1)$ consisting of the edges $\{c_1, b_1, b_2, b_3\}$.

Suppose we remove a branch b from a spanning tree T, then the tree T gets disconnected resulting in two trees (not spanning) T_1 and T_2. The sets of nodes in T_1 and T_2 define a partition of V. The corresponding cut is called the **fundamental cutset** of T with respect to branch b. For example, if we remove the branch b_3 from the tree T of Fig. 2(a) then we get trees T_1 and T_2 given by branches $\{b_1, b_2, b_5\}$ and $\{b_4\}$, respectively. The corresponding fundamental cutset $Q(b_3)$ consists of the edges $\{b_3, c_3, c_4, c_5, c_6\}$. Note that the subgraphs induced by the vertex sets of T_1 and T_2 are both connected. Cuts with this property are also called primary cuts [9].

Given a spanning tree T with branches $\{b_1, b_2, \dots, b_{n-1}\}$ and chords $\{c_1, c_2, \dots, c_{m-n+1}\}$, then the **fundamental circuit matrix** $B_f = [b_{ij}]_{(m-n+1)\times(m)}$ has one row for each chord and one column for each edge. The entry b_{ij} is defined as follows.

$b_{ij} = 1$, if $B(c_i)$ contains edge j
$= 0$, otherwise.

Arranging the rows of B_f such that the j^{th} row ($j \le m - n + 1$) corresponds to the fundamental circuit $B(c_j)$ and arranging the columns in the order $\{c_1, c_2, \dots, c_{m-n+1}, b_1, b_2, \dots, b_{n-1}\}$ we can write the B_f matrix as $B_f = [U | B_{ft}]$, where U is the unit matrix of size $(m - n + 1)$. For example, the B_f matrix with respect to the spanning tree T of Fig. 2(a) is given in (1).

In a similar manner the **fundamental cutset matrix** with respect to the tree T can be defined as $Q_f = [q_{ij}]_{(n-1)\times(m)}$. Q_f has $(n-1)$ rows, one for each fundamental cutset and one column for each edge. The entry q_{ij} is defined as

$q_{ij} = 1$, if $Q(b_i)$ contains edge j
$= 0$, otherwise.

Arranging the rows of Q_f such that the j^{th} row corresponds to *f*-cutset $Q(b_j)$ and the columns correspond to edges in the order $\{b_1, b_2, \dots, b_{n-1}, c_1, c_2, \dots, c_{m-n+1}\}$ the Q_f matrix can be

written as $Q_f = [U\,|Q_{fc}]$. For example, the Q_f matrix with respect to the tree T of Fig. 2(a) is given in (2).

	c_1	c_2	c_3	c_4	c_5	c_6	b_1	b_2	b_3	b_4	b_5
c_1	1	0	0	0	0	0	1	1	1	0	0
c_2	0	1	0	0	0	0	1	1	0	0	0
c_3	0	0	1	0	0	0	0	0	1	1	0
c_4	0	0	0	1	0	0	0	1	1	1	1
c_5	0	0	0	0	1	0	0	1	1	0	1
c_6	0	0	0	0	0	1	0	1	1	0	0

-- (1)

	b_1	b_2	b_3	b_4	b_5	c_1	c_2	c_3	c_4	c_5	c_6
b_1	1	0	0	0	0	1	1	0	0	0	0
b_2	0	1	0	0	0	1	1	0	1	1	1
b_3	0	0	1	0	0	1	0	1	1	1	1
b_4	0	0	0	1	0	0	0	1	1	0	0
b_5	0	0	0	0	1	0	0	0	1	1	0

--(2)

A subgraph (for example, a circuit or a cut) can be represented by a binary vector with m entries, one entry for each edge, and with an entry equal to 1 if the corresponding edge is present in the subgraph. Thus, rows of B_f and Q_f are the binary vectors representing the fundamental circuit and fundamental cutsets. For convenience, in the following we will use the same symbol $B(c_j)$ ($Q(b_j)$) to denote the set of edges in a fundamental circuit (cutset) as well as the corresponding binary vectors.

Proofs of the following dual results can be found in [14].

Theorem 1:

(a) If a cut contains the branches $\{b_1, b_2, \dots, b_j\}$ then the corresponding cut vector can be represented as $modulo\ 2$ addition of the vectors $Q(b_1), Q(b_2), \dots, Q(b_j)$. That is, the cut vector is equal to $Q(b_1) \oplus Q(b_2) \oplus \dots \oplus Q(b_j)$.

(b) If a circuit contains the chords $\{c_1, c_2, \dots, c_j\}$ then the corresponding circuit vector can be represented as $modulo\ 2$ addition of the vectors $B(c_1), B(c_2), \dots, B(c_j)$. That is, the circuit vector is equal to $B(c_1) \oplus B(c_2) \oplus \dots \oplus B(c_j)$. ■

Theorem 2 (***Orthogonality***): A circuit and a cut have an even number of common edges. ■

Theorem 3: $B_{ft} = Q_{fc}^t$, where Q_{fc}^t is the transpose of Q_{fc}. ■

The above properties can be verified using the B_f and Q_f matrices given in equations (1) and (2).

An ordered sequence $B(c_1), B(c_2), \dots, B(c_k)$ is a **circuit cover sequence or simply a *B*-sequence** of length k if

a) $\left[B(c_j) - c_j - \bigcup_{p=1}^{j-1} B(c_p)\right] \neq \emptyset, 2 \leq j \leq k$

b) $\bigcup_{p=1}^{k} B(c_p) = E - \{\text{chords not in the } B - \text{sequence}\}$

Note that for a given spanning tree and its *f*-circuits, there may be more than one *B*-sequence. For example for the fundamental circuits given in (1), following are the three *B*-sequences:

$$(1) B(c_1), B(c_3), B(c_5); \quad (2) B(c_4), B(c_1); \quad (3) B(c_6), B(c_1), B(c_4)$$

Note that the order in which the $B(c_j)$'s appear matters in the definition of *B*-sequences. Without the loss of generality assume that $B(c_1), B(c_2), \dots, B(c_k)$ is a *B*-sequence of length k. Let us define $S(c_j)$ as follows:

a) $S(c_1) = B(c_1) - c_1$

b) $S(c_j) = B(c_j) - c_j - \bigcup_{p=1}^{j-1} B(c_p), 2 \leq j \leq k$

Then the submatrix of the *f*-circuit comprised of the rows corresponding to $B(c_1), B(c_2), \dots, B(c_k)$ will have the structure shown in (3). An ordered sequence $Q(b_1), Q(b_2), \dots, Q(b_k)$ is a **cutset cover sequence or simply a *Q*-sequence** of length k if

a) $\left[Q(b_j) - b_j - \bigcup_{p=1}^{j-1} Q(b_p)\right] \neq \emptyset, 2 \leq j \leq k$

b) $\bigcup_{p=1}^{k} Q(b_p) = E - \{\text{branches not in the } Q - \text{sequence}\}$

Note that for a given spanning tree and its *f*-cutsets, there may be more than one *Q*-sequence. For example for the fundamental cutsets given in (2), following are the three *Q*-sequences.

$$(1) Q(b_4), Q(b_5), Q(b_3); \quad (2)\ Q(b_4), Q(b_5), Q(b_1), Q(b_2); \quad (3)\ Q(b_1), Q(b_2), Q(b_4)$$

Without the loss of generality assume that $Q(b_1), Q(b_2), \dots, Q(b_k)$ is *Q*-sequence of length k. Let us define $\hat{S}(b_j)$ as follows:

a) $\hat{S}(b_1) = Q(b_1) - b_1$

b) $\hat{S}(b_j) = Q(b_j) - b_j - \bigcup_{p=1}^{j-1} Q(b_p), 2 \leq j \leq k$

Then the submatrix of the *f*-cutset comprised of the rows corresponding to $Q(b_1), Q(b_2), \dots, Q(b_k)$ has a structure similar to (3) as shown in (4).

The following dual results are a consequence of the structures in (3) and (4).

Theorem 4:

(a) Given a *B*-sequence $B(c_1), B(c_2), \dots, B(c_k)$, let $B(c_{i_1}), B(c_{i_2}), \dots, B(c_{i_l})$ be a sub-sequence of this sequence then $S(c_{i_l}) \subseteq B(c_{i_1}) \oplus B(c_{i_2}) \oplus \dots \oplus B(c_{i_l})$.

(b) Given a *Q*-sequence $Q(b_1), Q(b_2), \dots, Q(b_k)$, let $Q(b_{i_1}), Q(b_{i_2}), \dots, Q(b_{i_l})$ be a subsequence of this sequence then $\hat{S}(b_{i_l}) \subseteq Q(b_{i_1}) \oplus Q(b_{i_2}) \oplus \dots \oplus Q(b_{i_l})$. ■

Deletion of an edge and **contraction** of an edge are dual operations. Here by contraction of an edge we refer to the operation of identifying the end vertices of the edge (short-circuiting the end vertices) and removing self loops that result from this short-circuiting.

Given a *B*-sequence, the submatrix of the B_f matrix that results after removing the rows that do not correspond to the chords in the *B*-sequence is called a ***B*-sequence matrix**. For example, the *B*-sequence matrix corresponding to the *B*-sequence $B(c_1), B(c_3), B(c_5)$ is shown in (5).

It can be shown that deletion of a row from B_f matrix corresponds to the deletion of the corresponding chord from the graph.

Given a *Q*-sequence, the submatrix of the Q_f matrix that results after removing the rows that do not correspond to the branches in the *Q*-sequence is called a ***Q*-sequence matrix**. For example, the *Q*-sequence matrix corresponding to the *Q*-sequence $Q(b_4), Q(b_5), Q(b_2), Q(b_1)$ is shown in (6).

It can be shown that deletion of a row from the Q_f matrix corresponds to contraction of the corresponding branch from the graph.

The following dual results are easy to verify.

Theorem 5:

a) The *B*-sequence matrix corresponding to a *B*-sequence is the fundamental circuit matrix of the graph that results after deleting chords that do not appear in the *B*-sequence.

b) The *Q*-sequence matrix corresponding to a *Q*-sequence is the fundamental cutset matrix of the graph that results after contracting branches that do not appear in the *Q*-sequence. ■

The incidence set of a vertex v in a graph is the set of vertices incident on that vertex. The incidence set of vertex v will be denoted by $INC(v)$. Each incident set is a cut of the graph. It is known that any set of $n-1$ incidence sets can be used to generate any cut in a graph. The incident vector $INC(v)$ of a vertex v is the binary vector of m entries with j^{th} entry being 1, if the j^{th} edge is in the corresponding incident set. The following result is a special case of Theorem 1(a). Note that we use $INC(v)$ to denote both the incidence set and the corresponding binary vector. **Incidence matrix** of $\boldsymbol{G}$ is the matrix of incidence vectors of G.

Theorem 6: The cut vector corresponding to the cut $(S, \bar{S})$ can be obtained as the $modulo\ 2$ addition of the incident vectors of the vertices in S as well as the $modulo\ 2$ addition of the incidence vectors of the vertices in $\bar{S}$. ■

A sequence of $INC(v_1), INC(v_2), \ldots, INC(v_k)$ is an **incident cover sequence or simply an $\boldsymbol{INC}$-sequence** of length k if

a) $\left[INC(v_j) - \bigcup_{p=1}^{j-1} INC(v_p)\right] \neq \emptyset, 2 \leq j \leq k$

b) $\bigcup_{p=1}^{k} INC(v_p) = E$

Given an INC-sequence, the submatrix of the incidence matrix consisting of the rows corresponding to the vertices in the INC-sequence has structure similar to the structure in (4).

The following result will be used in the proof of correctness of all the algorithms developed in the following sections. This is also the basis of the algorithmic framework given in [7].

Theorem 7 [14]: A graph is connected if and only if every cut of the graph contains at least one edge. ■

Finally, for convenience in presentation we shall call a logical graph **survivable** if the edges in the graph can be mapped into lightpaths in the physical topology so that a single physical edge failure does not cause the logical graph to become disconnected. The corresponding mappings will be called **survivable mappings**.

III. CIRCUIT-SMART: PRIMAL ALGORITHM FOR SURVIVABLE LOGICAL TOPOLOGY DESIGN

Given a logical topology G_L and physical topology G of an optical network, the SMART algorithmic framework given in [5] provides a methodology for finding survivable mappings of the edges of G_L into lightpaths in G. To begin with, let us call G_L the current graph. In general SMART consists of the following steps.

1. Search for a survivable subgraph of G_L. If no such subgraph is found, then terminate SMART unsuccessfully.
2. If a survivable subgraph is found then contract the edges of the subgraph.
3. If the current graph is a single vertex, then terminate SMART successfully. Otherwise, return to step 1 and use the contracted graph as the current graph.

The mappings of the edges considered by SMART provide a survivable subgraph of G_L. All the edges that were not mapped by SMART can be mapped arbitrarily without affecting the survivability of G_L. It can be shown that the subgraph chosen in step 1 above must be 2-edge connected for

$$\begin{array}{c|cccccc|cccc|cccc|cccc|cccc|cccc} & c_1 & \ldots & c_j & \ldots & c_k & & S(c_1) & & & & S(c_2) & & & & \ldots & & & & S(c_j) & & & & S(c_k) & & & \\ c_1 & 1 & \ldots & 0 & \ldots & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\ c_2 & 0 & \ldots & 0 & \ldots & 0 & \times & \times & \times & \times & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\ \vdots & \vdots & & \vdots & & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ \vdots & \vdots & & \vdots & & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ c_j & 0 & \ldots & 1 & \ldots & 0 & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & \\ c_k & 0 & \ldots & 0 & \ldots & 1 & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & 1 & 1 & 1 & 1 & \end{array} \quad \ldots(3)$$

$$\begin{array}{c|cccccc|cccc|cccc|cccc|cccc|cccc} & b_1 & \ldots & b_j & \ldots & b_k & & \hat{S}(b_1) & & & & \hat{S}(b_2) & & & & \ldots & & & & \hat{S}(b_j) & & & & \hat{S}(b_k) & & & \\ b_1 & 1 & \ldots & 0 & \ldots & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\ b_2 & 0 & \ldots & 0 & \ldots & 0 & \times & \times & \times & \times & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\ \vdots & \vdots & & \vdots & & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ \vdots & \vdots & & \vdots & & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ b_j & 0 & \ldots & 1 & \ldots & 0 & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & \\ b_k & 0 & \ldots & 0 & \ldots & 1 & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times & 1 & 1 & 1 & 1 & \end{array} \quad \ldots(4)$$

In the above × means 0 or 1.

$$\begin{array}{ccccccccc} c_1 & c_3 & c_5 & b_1 & b_2 & b_3 & b_4 & b_5 & \\ 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & \end{array} \quad --(5)$$

$$\begin{array}{ccccccccc} b_4 & b_5 & b_2 & b_1 & c_1 & c_2 & c_3 & c_4 & c_5 & c_6 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{array} \quad --(6)$$

the correctness of the algorithm. Since a circuit is a smallest 2-edge connected graph, usually in step 1 a circuit is selected for survivable mapping.

Now consider the graph G_L in Fig. 3. If the circuits are selected in the sequence C_1, C_2, C_3, C_4 and $\{e_6, e_{10}, e_{12}, e_{17}, e_{12}, e_{17}, e_{16}, e_{19}, e_{20}, e_{23}, e_3\}$ then these are fundamental circuits of the tree for which $\{e_1, e_8, e_{15}, e_{21}, e_6\}$ are chords. This observation is true for any choice of circuits selected and mapped by SMART. So our algorithm CIRCUIT-SMART starts with a set of fundamental circuits and a B-sequence constructed from these circuits. Since our interest is to guarantee survivability we add to G_L new edges in parallel to some of the edges in G_L whenever necessary. The new edge added in parallel to edge e of G_L will be denoted as e'. Both e and e' will be mapped into disjoint lightpaths in G. These edges will be called **protection edges.**

Algorithm CIRCUIT-SMART

Input: A 2-edge connected physical topology G, a 2-edge connected logical topology G_L, a spanning tree T of G_L, a set of fundamental circuits and a B-sequence $B(c_1), B(c_2), \ldots, B(c_k)$.

Output: A survivable logical graph G_L' containing G_L.

1. **For** $i = 1,2, \ldots, k$ **do**
   Map a maximum subset of edges in $S(c_k) \cup c_k$, into disjoint lightpaths in G. To all other edges in $S(c_k) \cup c_k$ add protection edges and map each edge and its protection edge into disjoint lightpaths in G.
   END FOR
2. Map all the chords not in the B-sequence into lightpaths in G arbitrarily.

END

Theorem 8: The graph G_L' of edges mapped by CIRCUIT-SMART forms a survivable logical graph.

Proof: We prove that G_L' is survivable by showing that after failure of any edge in the physical topology each cut in G_L' satisfies the condition in Theorem 7. Consider a cut in G_L'. If any edge in this cut is a protection edge e', then this edge and the corresponding edge e in G_L' are mapped by the algorithm into disjoint lightpaths. Hence, one of them will remain in the

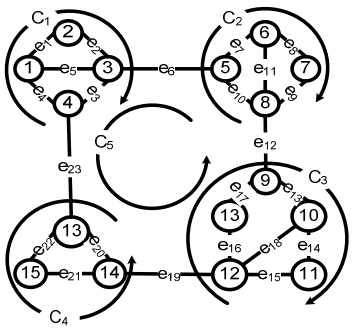


Figure 3. Illustration of SMART.

cut after a single physical edge failure, thereby satisfying the condition in Theorem 7. If there is no protection edge in the cut, then consider the branches in the cut. The cut must contain at least one branch of T because chords alone cannot form a cut. Note that each branch is in a unique $S(c_i)$. Let chord c_i be the chord with the smallest index in the B-sequence such that $S(c_i)$ contains one of the branches in the selected cut. If $S(c_i)$ contains two or more branches in the cut, then these branches are mapped by the algorithm into disjoint paths and so the cut will satisfy the condition of Theorem 7 after a single link failure. If $S(c_i)$ contains only one branch of the cut, say b, the cut must contain chord c_i because the cut and $B(c_i)$ must contain an even number of edges in common (Theorem 2). Since b and c_i are mapped by the algorithm into disjoint paths in the physical topology, one of these two edges will remain in the cut after a single edge failure, satisfying the condition of Theorem 7. Thus in all cases, the cut will satisfy the condition of Theorem 7 and so the graph G_L' is survivable. ■

The essential difference between SMART and CIRCUIT-SMART is that instead of searching for a survivable circuit in each step (as in SMART), CIRCUIT-SMART uses a set of fundamental circuits. Since not all edges in a $S(c_i)$ set can be mapped in a disjoint manner, we add protection edges appropriately. The longer the B-sequence, the more are the number of edges in the survivable logical subgraph. On the other hand a smaller B-sequence may increase the sizes of $S(c_i)$-sets and hence may result in more number of protection edges. These are considerations that must be taken into account while selecting the spanning tree.

IV. CUTSET-SMART: DUAL ALGORITHM

We now present algorithm CUTSET-SMART that is the dual of algorithm CIRCUIT-SMART. In the following a branch is **unmatched** if it is not in the given Q-sequence.

Algorithm CUTSET-SMART

Input: A 2-edge connected physical topology G, a 2-edge connected logical topology G_L, a spanning tree T of G_L, a set of fundamental cutsets and a Q-sequence $Q(b_1), Q(b_2), \ldots, Q(b_k)$.

Output: A survivable logical graph G_L'' containing G_L.

1. **For** i $= 1,2, \ldots,$ k **do**
   Map a maximum subset of edges in $\hat{S}(b_k) \cup b_k$ into disjoint lightpaths in G. To all other edges in $\hat{S}(b_k) \cup b_k$ add protection edges and map each edge and its protection edge into disjoint lightpaths in G (see [16]).

END FOR

2. To each unmatched branch b add a protection edge b' and map them into disjoint lightpaths in G.

END

Let G_L'' be the graph of logical edges (including protection edges) mapped by CUTSET-SMART.

Theorem 9:

a) The graph G_L'' with the mappings generated by CUTSET-SMART is survivable.

b) The graph obtained from G_L'' by contracting the branches not in the Q-sequence is survivable.

Proof: a) Consider a cut in G_L''. If any edge in this cut is a protection edge then this edge and the corresponding edge in G_L are mapped by the algorithm into disjoint lightpaths. Hence one of them will remain in the cut after a single physical edge failure, thereby satisfying the condition in Theorem 7. If there is no protection edge in the cut, then consider the branch b_j in the cut that has the highest index in the Q-sequence Then by Theorem 4(b) the set $\hat{S}(b_j)$ will be in the cut. Since the branch b_j and the edges in the set $\hat{S}(b_j)$ are mapped in disjoint manner by the algorithm, the cut will contain at least one edge after a physical edge failure, thereby satisfying the condition of Theorem 7. Thus G_L'' is survivable.

b) The graph obtained from G_L'' by contracting the branches not in the Q-sequence has no cut that contains the contracted tree branches and the corresponding protection edges. The result follows from the earlier part of the proof of (a). ■

Note that Theorem 9(b) is the dual of the result that graph G_L' generated by CIRCUIT-SMART is survivable.

A closer look at the above proof will show that in step 1 of CUTSET-SMART, it is sufficient to map in disjoint manner each branch b_i with some chord in the set $\hat{S}(b_i)$. This results in the following algorithm CUTSET_SMART_ SIMPLIFIED.

Algorithm CUTSET_SMART_SIMPLIFIED

Input: A 2-edge connected physical topology G, a 2-edge connected logical topology G_L, a spanning tree T of G_L, a set of fundamental cutsets and a Q-sequence $Q(b_1), Q(b_2), \ldots, Q(b_k)$.

Output: A survivable logical graph G_L'' containing G_L.

1. **For** $i = 1,2, \ldots, k$ **do**
   Map b_i in disjoint manner with some chord in set $\hat{S}(b_i)$. If this is not possible for any chord in $\hat{S}(b_i)$ then add a protection edge to one of these chords and map the chord and the protection edge in disjoint manner.

END FOR

2. To each unmatched branch b, add a protection edge b' and map them as disjoint lightpaths in G.
3. Map all the unmapped logical edges arbitrarily.

END

The above algorithm requires finding disjoint mappings for only certain pairs of vertices in the physical topology. Using a result in [6] we have the following theorem.

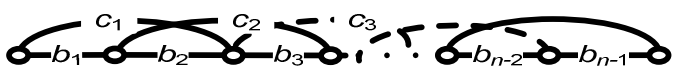


Figure 4. A survivable network structure.

Theorem 10: Given any Q-sequence of length k, algorithm CUTSET_SMART_SIMPLIFIED finds a survivable mapping of a logical topology with at most $n-k-1$ protection edges, if the physical topology is 3-edge connected. ■

The main reason for the above result is that any pair of edges $\{(s_1,t_1),\ (s_2,t_2)\}$ in G_L can be mapped into disjoint lightpaths, if the physical topology is 3-edge connected.

The proof of the Theorem 9 shows that every cut obtained from G_L'' by contracting the branches not in the Q-sequence will contain at least $Min\{|\hat{S}(b_i)|+1, i=1,2,\dots,k\}$ edges after a single edge failure in the physical topology. This property of CUTSET-SMART will be of great help in protecting the logical topology when multiple edge failures occur in the physical topology. This property is not true in the case of algorithm CUTSET_SMART_SIMPLIFIED though it is computationally superior to CUSET-SMART and is likely to require less number of protection edges. An interesting consequence of algorithm CUTSET_SMART_SIMPLIFIED is the following.

Theorem 11: The structure shown in Fig. 4 is survivable, if the physical topology is 3-edge connected. ■

An interesting application of this result is as follows. Note that protection edges are used in the algorithms of sections III and IV to guarantee survivability. Consider a set of edges that form a path and require protection edges, then it can be shown that we can add new logical edges $c_1, c_2, \dots \dots$ as in Fig. 4, instead of protection edges (that is, new parallel logical edges) and guarantee survivability.

Note that we have not been able to prove a property similar to the one in Theorem 10 in the case of CIRCUIT-SMART. Nor has it been possible for us to find a simplified version of CIRCUIT-SMART akin to CUTSET_SMART_SIMPLIFIED.

V. Efficient Heuristics and Minimizing Number of Protection Edges

In cutset based algorithms, to guarantee survivability we add protection edges in parallel to branches (unmatched branches) that are not in the Q-sequence. See step (2) in these algorithms. To minimize the number of such protection edges we suggest two heuristics.

A 2-Phase method: In this method we replace step (2) of CUTSET-SMART and CUTSET_SMART_SIMPLIFIED by the following:

"Apply CIRCUIT-SMART on the graph obtained from G_L by contracting the matched branches (branches in the Q-sequence)."

We have not been able to show that the above heuristic will guarantee survivability in certain scenarios. However, we expect this to guarantee survivability except in the case of certain combinations of logical edge failures.

Maximum Matching Based Heuristic:

1. **For** $i = 1,2,\dots,k$ **do**
   Map in disjoint way b_i and some chord in the set $\hat{S}(b_k)$.
   END FOR
2. Construct a bipartite graph (X,Y) where each vertex in X represents an unmatched branch and each vertex in Y represents an unmatched chord. The bipartite graph has an edge (x,y) if the fundamental cutset with respect to branch x contains the chord y.
3. Find a maximum matching in this bipartite graph using any standard maximum matching algorithm [14].
4. For each edge in the maximum matching, find disjoint mappings for the corresponding pair of branch and chord.
5. For all unmatched branches (those not matched in step 1 or 2) add protection edges as in step (2) of CUTSET_SMART.

END

The above heuristic is not expected to guarantee survivability. However, we expect this to guarantee survivability except in the case of special cuts in G_L.

The performance of these heuristics will be studied through simulation (Section VII).

VI. Incidence-SMART

Incident sets are special cases of cuts. So an algorithm similar to CUTSET-SMART can be designed. Instead of explicitly starting with a spanning tree with some new logical edges, we present an algorithm which reflects the unified framework in terms of INC-sets defined in section II. Note that for any INC-sequence there is at least one vertex that is not in the sequence. Let us call one such vertex as datum.

In the following algorithm the given G_L will be the initial current graph.

Algorithm INCIDENCE-SMART

Input: A 2-edge connected physical topology G, a 2-edge connected logical topology G_L and INC -sequence $INC(v_1)$, $INC(v_2), \dots, INC(v_k)$.

Output: A survivable logical graph G_L'' containing G_L.

1. **For** $i = 1,2,\dots k$ **do**
   1) If vertex v_i has degree greater than or equal to 2 in the current graph, then map all the edges incident on v_i into disjoint lightpaths in G.
   2) If the degree of v_i in the current graph is one, then add a new logical edge connecting v_i to the datum vertex. Then map this new edge and the only edge incident on v_i into disjoint lightpaths.
   3) If degree of v_i in the current graph is zero add two new parallel logical edges connecting v_i to the datum vertex. Then map these two edges into disjoint lightpaths in G.
   END FOR

END

Theorem 12: Algorithm INCIDENCE-SMART provides a survivable mapping of the edges of a graph G_L'' that contains the given logical graph G_L.

Proof: Consider any cut $(S,\bar{S})$ in G_L. Let S be the partition of the cut that does not contain the datum vertex. Consider the vertex v in S that has the highest index in the INC-sequence. Then in the current graph at the step when v is considered by the algorithm it will not be adjacent to any vertex in S. So, according to the algorithm v will be connected to at least two vertices in $\bar{S}$, and the corresponding edges connecting S and $\bar{S}$ are mapped into disjoint lightpaths, guaranteeing that at least one of these edges will remain in the cut after a single edge failure in G thereby satisfying the condition of Theorem 7.

Since this is true for all cuts, the mappings generated by the algorithm are survivable. ■

VII. Simulation Study and Results

To compare the performance of the proposed algorithms, simulation studies were conducted using VC++ 8.0. For simulation studies, random logical topologies with varying number of nodes and degree were generated. The physical topologies were regular topologies with degree 4, constructed using a procedure originally given by Harary and described in [14]. The number of nodes in the physical topologies was set to 100, and 200 nodes ($|N|$). The logical topologies were generated randomly with average degrees 2.5, 3.0, 3.5 and 4.0. The nodes in the logical topologies were a subset of the physical nodes and number of logical nodes in the logical topology was set to $0.75 \times |N|$. For each case, 40 physical and 25 logical topologies were generated, providing a total of 1000 logical-physical topology pairs for comparison. The topologies were subjected to further processing, only if the topologies met the connectivity requirements.

To find the maximum number of logical links that could be mapped in a mutually disjoint manner, a procedure described in [16] was used. To find mutually disjoint mappings of a pair of logical links, the algorithm given in [6] was used. Fundamental circuits and cutsets were found using procedures given in [14] and were part of the preprocessing phase. The survivability of a logical topology was tested by picking a physical link, removing all the logical links which used this physical link in their mapping, and checking if the resulting logical topology was connected. This test was repeated for every physical link

The statistics of interest were survivability success rate (that is, the number of logical topologies for which survivable mappings could be found), protection capacity (measured as average number of protection links added to a logical topology to make it survivable) and the execution time of the algorithms.

We now make some general observations on the performance of these algorithms based on the trends that we noticed during the simulations.

Tables II-VII summarize the results. Tables II and V provide the number of successful survivable logical-physical topology pairs. It can be seen that we were able to find survivable mappings for 100% of the topologies (see Table I for legend) as expected in the case of $M1, M2, M3$ and $M7$. Also, we would like to point out that in the case of $M4$ and $M6$ we were able to achieve 100% survivability in all the tests but we have not been able to prove that this result holds in general.

Tables III and VI show a general trend that the number of protection edges required in the case of circuit based methods is considerably less than the number required by cutset based methods. The number of protection edges required by $M4$, which is obtained by integrating $M1$ and $M3$, is less than the number for $M1$. Note that in the case of cutset based methods, for a given value of n the number of edges in a spanning tree does not change. Since $M3$ maps only $n-1$ pairs of edges, the computation time for this method does not change very significantly with a change in average degree.

$M3$ is the best in terms of execution time. $M7$ is the next best. $M7$ also does very well in terms of number of protection edges required.

Summarizing, if protection capacity is an issue of concern, then the two phase algorithm $M4$ is a good choice but it does not guarantee survivability. If computational time is an issue of primary interest, then $M3$ and $M7$ are good choices. Finally, if guarantee of survivability is also an issue of interest then $M7$ is the best choice.

VIII. Summary and Conclusions

Duality plays a significant role in optimization theory, particularly in discrete optimization on graphs and networks. Circuits and cuts in graphs are dual concepts. Similarly, deletion and contraction of an edge are dual graph operations. Most often, for a primal algorithm based on circuits there is a dual algorithm based on cuts for the same problem. The primal and dual algorithms possess certain characteristics that make one superior to the other depending on the application. The SMART algorithm [5] for the survivable logical topology design problem is based on circuits. This is a novel and very signifcant contribution to the problem. The question then arises whether there exists a dual methodology based on cuts. In this paper we have investigaed this question and developed certain new dual algorithms and insightful results.

First, we reviewed several results that highlight the dual relationship between circuits and cuts and the dual graph operations of deletion and contraction of an edge. We also introduced certain new concepts and results that form the foundation of the methodologies developed in the rest of the paper. We then presented the primal algorithm CIRCUIT-SMART (similar to SMART) and algorithm CUTSET-SMART that is dual of CIRCUIT-SMART and unified proofs of correctness of these algorithms. Our investigation has provided much insight into the structural properties of solutions to this problem and the structure of survivable logical graphs (Theorems 10 and 11). Specifically, we presented a highly simplified version of CUTSET-SMART that always provides a survivable mapping as long as the physical topology is 3-edge connected. We presented a survivable logical topology structure that can be embedded into a survivable toplogy (Theorem 11) without affecting survivability. We also presented algorithm INCIDENCE-SMART that uses incidence sets. Two efficient heursitcs, one based on maximum matching and a 2-phase method that combines both the primal and dual algorithms, were also presented. To guarantee survivability, our algorithms add new logical edges called protection edges, whenever needed. Simulations comparing the different algorithms in terms of computational time, protection capacity and survivability success rate are also provided.

We wish to add that though all our algorithms are formulated in terms of fundamental circuits or fudamental cutsets, they can be presented in a general form as in the original SMART algoirthm of [5]. Such a general form would require a search of a spanning tree whose fundamental circuits or cutsets can be mapped in a survivble manner. In our simulations, we have not compared our algoirthms with the

original SMART algorithm for two reasons. First, SMART will require signficantlly higher execution times because it searches for survivable circuits and so the comparison will not be fair. Secondly, SMART does not add protection edges to gaurantee survivability.

As we stated in section VII, if protection capacity is an issue of concern, then the two phase algorithm, CUTSET_SMART_SIMPLIFIED combined with CIRCUIT_SMART, is a good choice but it does not guarantee survivability. If computational time is an issue of primary interest, then CUTSET_SMART_SIMPLIFIED and INCIDENCE_SMART are good choices. Finally, if guarantee of survivability is also an issue of interest then INCIDENCE_SMART is the best choice.

The performance of our algorithms depends on the choice of the spanning tree and the resulting *B*- and *Q*- sequences. Research is in progress on approaches for selecting appropriate spanning trees. We believe that our work studying the problem from diferent perspectives has advanced the state of the art and thrown much insight into the problem.

REFERENCES

[1] N. Ghani, S. Dixit, and T. Wang, "On IP-over-WDM Integration," *IEEE Comm. Magazine*, March 2000.

[2] I. Chlamtac, A. Ganz, and G. Karmi, "Lightpath Communications: An Approach to High Bandwidth Optical WDM," *IEEE Trans. on Comm.*, vol. 40, no. 7. July 1992

[3] O. Crochat, J. Boudec, and O. Gerstel, "Protection Interoperability for WDM Optical Networks," *IEEE/ACM Trans. on Networking*, vol. 8, no. 3, June 2000.

[4] T. Stern, and K. Bala, Multiwavelength Optical Networks: A Layered Approach, Addison-Wesley Longman Boston, MA, 1999.

[5] M. Kurant, and P. Thiran, "On Survivable Routing of Mesh Topologies in IP-over-WDM Networks," *Proc. of INFOCOM* 2005.

[6] Y. Perl and Y. Shiloach, "Finding two disjoint paths between two pairs of vertices in a graph," *J. of the ACM*, 25(1):1-9, January 1978.

[7] E. Modiano and A. Narula-Tam, "Survivable Routing of Logical Topologies in WDM Networks," *Proc. of INFOCOM* 2001.

[8] E. Modiano and A. Narula-Tam, "Survivable Lightpath Routing: A New Approach to the Design of WDM-Based Networks," *IEEE J. on Sel. Areas in Comm.*, vol. 20, no. 4, May 2002.

[9] A. Todimala and B. Ramamurthy, "A Scalable Approach for Survivable Virtual Topology Routing in Optical WDM Networks," *IEEE J. on Sel. Areas in Comm*., Vol. 25, No. 6, Aug 2007.

[10] F. Ducatelle and L. Gambardella, "FastSurv: A New Efficient Local Search Algorithm for Survivable Routing in WDM Networks," *Proc. of GLOBECOM 2004*.

[11] R. Shenai and K. Sivalingam, "Hybrid Survivability Approaches for Optical WDM Mesh Networks," *J. of Lightwave Technology*, Vol. 23, No. 10, Oct 2005.

[12] M. Kurant and P. Thiran, "Survivable Mapping Algorithm by Ring Trimming (SMART) for large IP-over-WDM networks," *Proc. of BroadNets 2004*.

[13] M. Kurant and P. Thiran, "Survivable Routing of Mesh Topologies in IP-over-WDM Networks by Recursive Graph Contraction," *IEEE J. on Sel. Areas in Comm.*, vol. 25, no. 5, pp. 922-933, June, 2007.

[14] K. Thulasiraman and M. N. S. Swamy, GRAPHS: Theory and Algorithms, Wiley-Inter-science 1992.

[15] R. Ahuja, T. Magnanti, and J. Orlin, Network Flows: Theory, Algorithms, and Applications, Prentice Hall, 1993.

[16] J. Kleinberg. Approximation Algorithms for Disjoint Paths Problems. Ph.D Thesis, Dept. of EECS, MIT (1996).

Table I. Legend

Circuit-SMART	$M1$	Cutset-SMART	$M2$
Cutset_SMART_SIMP	$M3$	2 Phase Method (with Cutset_SMART_SIMP)	$M4$
Max-Mat. + Parallel protection edges	$M5$	Max-Mat. + Circuit SMART	$M6$
Incidence-SMART	$M7$	Logical Degree	LD
Physical Degree	PD	Method Number	M

Table II. Survivable Logical-Physical Topology Pairs (|N| = 100, PD = 4)

LD/M	2.5	3.0	3.5	4.0
$M1$	1000	1000	1000	1000
$M2$	1000	1000	1000	1000
$M3$	1000	1000	1000	1000
$M4$	1000	1000	1000	1000
$M5$	460	544	654	748
$M6$	1000	1000	1000	1000
$M7$	1000	1000	1000	1000

Table III. Ave. No. of Protected Links (|N| = 100, PD = 4)

LD/M	2.5	3.0	3.5	4.0
$M1$	44.65	23.84	15.46	10.72
$M2$	56.66	45.95	45.24	44.46
$M3$	56.19	43.55	35.10	27.88
$M4$	18.79	6.01	3.85	2.31
$M5$	55.22	36.72	18.42	3.85
$M6$	38.97	3.15	1.23	1.32
$M7$	55.00	36.00	17.20	8.30

Table IV. Time per Logical-Physical Topology-Pair (sec) (|N| = 100, PD = 4)

LD/M	2.5	3.0	3.5	4.0
$M1$	5.302	3.702	2.965	2.521
$M2$	0.427	0.864	1.869	3.042
$M3$	0.352	0.440	0.529	0.592
$M4$	5.632	2.404	1.943	1.647
$M5$	1.382	1.207	1.069	0.922
$M6$	4.115	0.863	0.770	0.874
$M7$	0.489	0.508	0.520	0.794

Table V. Survivable Logical-Physical Topology Pairs (|N| = 200, PD = 4)

LD/M	2.5	3.0	3.5	4.0
$M1$	1000	1000	1000	1000
$M2$	1000	1000	1000	1000
$M3$	1000	1000	1000	1000
$M4$	1000	1000	1000	1000
$M5$	480	608	701	750
$M6$	1000	1000	1000	1000
$M7$	1000	1000	1000	1000

Table VI. Ave. No. of Protected Links (|N| = 200, PD = 4)

LD/M	2.5	3.0	3.5	4.0
$M1$	103.69	62.04	44.66	28.79
$M2$	114.64	96.30	88.57	79.94
$M3$	113.30	89.83	71.90	61.18
$M4$	40.53	15.12	9.31	6.66
$M5$	111.7	75.02	40.28	8.71
$M6$	77.34	8.91	4.94	3.48
$M7$	111.00	73.00	36.00	13.33

Table VII. Time per Logical-Physical Topology-Pair (sec) (|N| = 200, PD = 4)

LD/M	2.5	3.0	3.5	4.0
$M1$	50.43	35.098	28.713	21.211
$M2$	2.189	5.794	11.883	25.213
$M3$	1.601	1.982	2.526	3.022
$M4$	39.993	18.796	14.834	13.388
$M5$	10.462	8.965	7.811	6.596
$M6$	27.542	6.747	4.965	6.131
$M7$	2.117	2.125	2.195	3.328