1+N Protection Against Multiple Link Failures in Mesh Networks

Ahmed E. Kamal
Dept. of Electrical and Computer Eng., Iowa State University, Ames, IA 50011, U.S.A.
E-mail: kamal@iastate.edu

***Abstract*—In [1], the author presented a 1+N protection strategy against single link failures using a network coding approach on p-Cycles. In this paper, we extend this approach to protect against multiple link failures. For the network to be protected against M link failures, M p-Cycles are used. The connections sharing a certain cycle must have link disjoint paths, and they encode their transmitted and received data units on two counter rotating half cycles on each of the two cycles. To recover from m link failures using M cycles, where $1 \leq m \leq M$, the data units are encoded on the cycles in such a way that each node affected by failures should recover m linearly independent combinations of the m units affected by the failures. To illustrate the concept, we show how to protect against two link failures, and describe in detail the encoding and decoding processes.**

I. Introduction

Predesigned Protection techniques provide survivable operation for optical networks by reserving bandwidth in advance. Therefore, when upon failures, pre-provisioned backup paths are used to reroute the traffic affected by the failure. These techniques include the 1+1 protection, in which traffic of a circuit is transmitted on two link disjoint paths, and the receiver selects the stronger of the two signals; 1:1 protection, which is similar to 1+1, except that traffic is not transmitted on the backup path until failure takes place; 1:N protection, which is similar to 1:1, except that one path is used to protect N paths; and M:N, where M protection paths are used to protect N working paths. The 1:1, 1:N and M:N techniques involve the management plane to detect failures, and the control plane to route against failures, which can prolong the failure recovery time. The concept of p-Cycles was introduced in [2], [3] to provide 1:N protection to connections with the same transport capacity, with an efficiency that approaches that of the optimal 1:N protection in mesh networks.

Recently, the author introduced a new concept for survivability, namely, 1+N protection in [1]. The technique is based on using a bidirectional p-Cycle to protect a number of link disjoint connections, and using network coding [4] to transmit modulo 2 sums of the connections signals on the cycle. A failure of any link on a primary path can be recovered from by using a decoding operation of the signals transmitted on the p-Cycle. This strategy was introduced to protect against a single link failure.

While single link failures are the most common type of failures in optical networks, there are situations in which multiple links may fail simultaneously. For example, when a number of fibers run through the same duct and the entire duct is subject to failure, all fibers fail simultaneously[1]. This is usually referred to as a Shared Risk Link Group (SRLG) [5]. While protection against multiple failures is more challenging, as compared to protection against a single failure, it has started to receive attention from researchers. Reference [6] compared between path protection and path rerouting under dual failures using optimal ILP models. Reference [7] considers network reconfiguration for protection against two link failures, and finds that this can result in up to 95% recovery from second failures. In [8] the same problem was considered for multiple sequential link failures but for connection protection, and two algorithms were introduced. In [9] three loopback strategies for link protection against two sequential link failures were introduced, and heuristics were developed. Reference [10] developed optimal implementations for one of the algorithms in [9] considering both dedicated link protection, and shared link protection. The problem of path protection against single and multiple link failures, as well as single node failures is considered in [11]. A matrix-based formulation for computing, and reducing the spare capacity allocation was developed. The Successive Survivable Routing heuristic was introduced, which iteratively routes and updates backup paths such that spare capacity allocation is minimized.

In this paper we extend our 1+N protection strategy in [1] to protect against multiple link failures. The extension is based on using multiple p-Cycles. However, the encoding and decoding operations for protecting against multiple failures are different from the single failure case. Moreover, certain conditions must be satisfied for data affected by failures to be recovered.

The rest of the paper is organized as follows. In Section II we introduce the strategy by which 1+N protection can be extended to protect against multiple link failures. The cycle and connection arrangements will be introduced, as well as the encoding and decoding operations. Data recovery algorithm, and sufficient and necessary conditions for recovery from multiple link failures will also be presented. In Section III we illustrate the details of applying this technique to protect against two failures, and Section IV concludes the paper with some remarks. For the sake of easy reference, we list the symbols used in this paper in Table I.

This research was supported in part by grant CNS-0626741 from the National Science Foundation.

[1]Another scenario is when a number of links, e.g., lightpaths, share a fiber.

TABLE I
LIST OF SYMBOLS

Symbol	Meaning
N	number of connections
M	total number of failures to be protected against
$\mathcal{S}$, $\mathcal{T}$	two classes of communicating nodes, such that a node in $\mathcal{S}$ communicates with a node in $\mathcal{T}$
S_i, T_j	nodes in $\mathcal{S}$ and $\mathcal{T}$, respectively
d_i, u_j	data units sent by nodes S_i and T_j, respectively
$\mathbf{C}_k$	kth bidirectional cycle used for protection
$\mathbf{T}_k$, $\mathbf{R}_k$	clockwise and counterclockwise unidirectional subcycles of $\mathbf{C}_k$
$\vec{\alpha}_k$	encoding vector used on cycle $\mathbf{C}_k$
n_k	number of connections protected by cycle $\mathbf{C}_k$
$\mathcal{S}_k$, $\mathcal{T}_k$	nodes in connections protected by cycle $\mathbf{C}_k$, such that $\mathcal{S}_k \subset \mathcal{S}$, and $\mathcal{T}_k \subset \mathcal{T}$
$\mathbb{N}$	set of connections to be protected
N_i	connection to be protected, such that $N_i \in \mathbb{N}$
$\mathbb{C}$	set of cycles to be used to protect $\mathbb{N}$

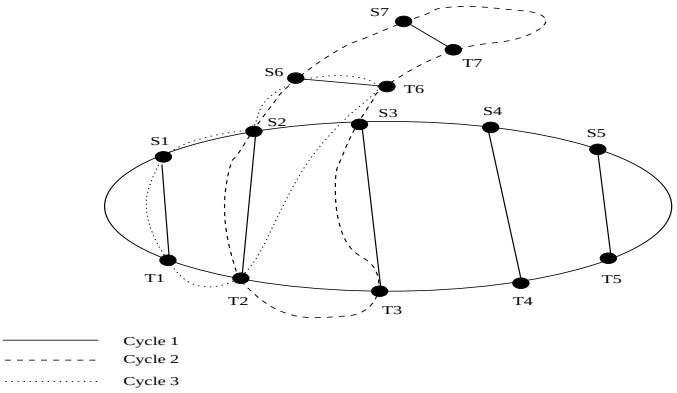

Fig. 1. An example of a network protected against a maximum of 3 failures.

II. 1+N PROTECTION AGAINST MULTIPLE FAILURES

In this section, we extend the procedure in [1] to provide protection against multiple failures.

A. Connections

1) There are N bidirectional connections between N pairs of nodes. The nodes in different connections need not be disjoint. That is, connections between the same pair of nodes, multicast connections, or many-to-one connections can be present. A node which participates in multiple connections is logically replicated, and different copies are treated as independent nodes.
2) Nodes are partitioned into two classes: $\mathcal{S}$ and $\mathcal{T}$, where node S_i in $\mathcal{S}$ sends d_i units to nodes in $\mathcal{T}$, and node T_j in $\mathcal{T}$ sends u_j units to nodes in $\mathcal{S}$. The enumeration of nodes in $\mathcal{S}$ and $\mathcal{T}$ is performed using the procedure in [1], and is explained below.
3) Primary paths used by connections to carry different data streams must be disjoint.
4) Node $S_i \in \mathcal{S}$ transmits data units d_i on the primary paths to the corresponding receivers, while node $T_i \in \mathcal{T}$ transmits data units u_i on the primary paths to corresponding receivers. The data units are fixed and equal in size for all connections. Assume that the packet length is s bits.

B. Protection Cycles

Protection is achieved by transmitting linear functions of data units on cycles, which are received by all nodes. In the case of failure, data affected by failures on primary paths can be recovered from the data transmitted on the cycles. It is to be noted that all addition operations are modulo 2 operations, i.e., bitwise XOR operations.

The following outlines how the cycles are constructed, and how data is encoded on such cycles.

1) For a network to be protected against a maximum of M link failures, it employs M cycles, which are referred to by $\mathbf{C}_k$ for $k = 1, 2, \ldots, M$. Protection cycle $\mathbf{C}_k$ passes through all nodes in $\mathcal{S}_k \subseteq \mathcal{S}$ and $\mathcal{T}_k \subseteq \mathcal{T}$ (in any arbitrary order), where nodes in $\mathcal{S}_k$ communicate bidirectionally with the nodes in $\mathcal{T}_k$. Note that $\cup_{k=1}^{M} \mathcal{S}_k = \mathcal{S}$, and $\cup_{k=1}^{M} \mathcal{T}_k = \mathcal{T}$. The $\mathcal{S}_k$ and $\mathcal{S}_l$ are not necessarily disjoint for $l \neq k$. The m cycles must be link disjoint, and they must also be link disjoint from the primary paths of the connections they protect.
2) The sequence of nodes in $\mathcal{S}_k$ is arbitrarily selected to be in the clockwise direction, and the sequence of nodes in $\mathcal{T}_k$ to be in the counter-clockwise direction. The nodes are enumerated such at an arbitrary node in $\mathbf{C}_k$ is labeled S_1. Proceeding in the clockwise direction, if a node has not been accounted for, it will be the next node in $\mathcal{S}_k$, and using increasing indices for S_i. Otherwise, it will be in $\mathcal{T}$, and using decreasing indices for T_i. Therefore, node T_1 will always be the one next to node S_1 in the counter-clockwise direction. The example in Figure 2 shows how ten nodes, in five connections are assigned to $\mathcal{S}$ and $\mathcal{T}$.
3) Each of the M cycles consists of two counter rotating directional cycles, which we refer to as $\mathbf{T}_k$ and $\mathbf{R}_k$ half cycles of cycle $\mathbf{C}_k$. We choose $\mathbf{T}_k$ and $\mathbf{R}_k$ to be in the clockwise, and counter-clockwise directions, respectively. Newly generated d_i and u_i data units are encoded and then transmitted on $\mathbf{T}_k$, and will be referred to as d_i^t and u_i^t. Data units received on the working paths are transmitted, after encoding, on the $\mathbf{R}_k$ half cycle.
4) A connection can be selectively protected against any number, m, of link failures where $1 \leq m \leq M$. A connection that is protected against m failures must be part of at least m protection cycles. The network in Figure 1 is protected against a maximum of 3 failures. Node pairs $(S4, T4)$, $(S5, T5)$, and $(S7, T7)$ are protected against a single link failure, node pairs $(S1, T1)$, $(S3, T3)$ and $(S6, T6)$ are protected against 2 failures, while node pair $(S2, T2)$ is protected against 3 failures.
5) Similar to the 1+N protection against single link failure, in addition to transmitting new data units on the working paths, sources of data units use a linear encoding function to transmit those data units on $\mathbf{T}_k$ cycles. Similarly, received data units are transmitted on $\mathbf{R}_k$ cycles. However, unlike the single failure protection

scheme, instead of just simply adding the data units using modulo 2 addition (XOR operation), d_i and u_j data units are transmitted on cycle $\mathbf{C}_k$ using a vector of integer scalar coefficients $\vec{\alpha}^k$, which is given by

$$\vec{\alpha}^k = (\alpha_1^k, \alpha_2^k, \ldots, \alpha_{n_k}^k) \tag{1}$$

where n_k is the number of connections protected by cycle $\mathbf{C}_k$. The encoding of d_i and u_j data units using $\vec{\alpha}^k$ will be described in the next section.

C. Data Encoding Operation

Transmissions occur in rounds, such that d_i^t data units which are encoded together and transmitted on a cycle must belong to the same round. Rounds can be started by the S_1 node, and are then followed by other nodes. All nodes in $\mathcal{S}$ and $\mathcal{T}$ must keep track of round numbers. The same round number conditions apply to rounds in which sums of encoded u_i^t data units are transmitted, as well as rounds for transmitting functions of d_i^r, and functions of u_i^r data units.

The encoding is implemented as follows, while noting that all additions (including those used to implement the multiplication operation) are bit-wise XOR operations, and that round numbers are observed:

1) Node S_i in $\mathcal{S}$ will do the following:
   a) It adds the following to the signal received on $\mathbf{T}_k$:
      i) Newly generated d_i^t units after multiplying them by α_i^k. This adds new data units to $\mathbf{T}_k$.
      ii) Data unit u_j^t which it received on the primary path from T_j after multiplying it by α_j^k. This removes this data unit, which was added by T_j, from the signal on $\mathbf{T}_k$.
   b) It also adds the following to the signal received on $\mathbf{R}_k$:
      i) Data unit d_i^r, which it transmitted in an earlier round, after multiplying it by α_i^k, which will remove this product from the signal on $\mathbf{R}_k$.
      ii) Data unit u_j^r, which it received on the primary path from T_j, after multiplying it by α_j^k, hence adding it to the signal on $\mathbf{R}_k$.
2) Similarly, node T_i will do the following:
   a) It adds the following to the signal received on $\mathbf{T}_k$:
      i) Newly generated u_i^t units after multiplying them by α_i^k. This adds new data units to $\mathbf{T}_k$.
      ii) Data unit d_j^t which it received on the primary path from S_j after multiplying it by α_j^k. This removes this data unit, which was added by S_j, from the signal on $\mathbf{T}_k$.
   b) It also adds the following to the signal received on $\mathbf{R}_k$:
      i) Data unit u_i^r, which it transmitted in an earlier round, after multiplying it by α_i^k, which will remove this product from the signal on $\mathbf{R}_k$.
      ii) Data unit d_j^r, which it received on the primary path from S_j, after multiplying it by α_j^k, hence adding it to the signal on $\mathbf{R}_k$.

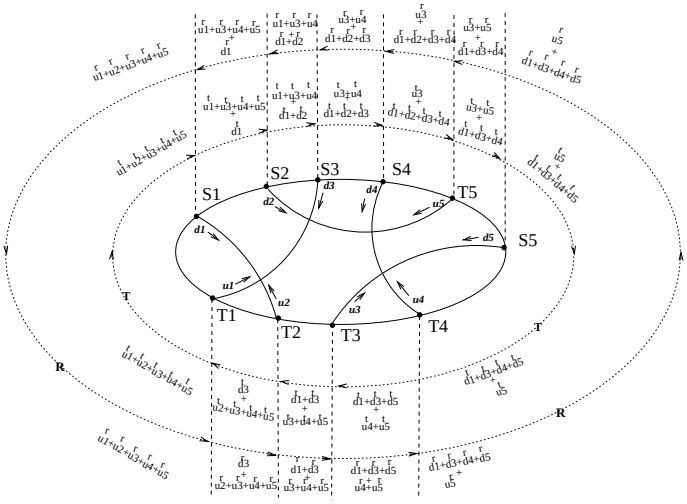

Fig. 2. An example of five nodes being protected by one cycle [1].

Figure 2 (taken from [1]) shows five connections protected by one cycle, and $\vec{\alpha}^1 = \vec{e}$, i.e., a vector of 1s.

The above procedure, as applied to one cycle $\mathbf{C}_k$ only, with the exception of using the $\vec{\alpha}^k$ vector, is similar to that in [1]. However, what is new here is the use of multiple cycles to protect against multiple link failures, and, as will be explained below, the selection of $\vec{\alpha}^k$ to satisfy certain properties, which is central to providing protection against multiple failures.

According to the above operations, node T_j receives on $\mathbf{T}_k$ and $\mathbf{R}_k$ the the following signals, respectively:

$$\alpha_i^k d_i^t + \sum_{l, S_l \in \mathcal{S}'_k} \alpha_l^k d_l^t + \sum_{l, T_l \in \mathcal{T}'_k} \alpha_l^k u_l^t \tag{2}$$

$$\alpha_j^k u_j^r + \sum_{l, S_l \in \mathcal{S}'_k} \alpha_l^k d_l^r + \sum_{l, T_l \in \mathcal{T}'_k} \alpha_l^k u_l^r \tag{3}$$

for some $\mathcal{T}'_k \subset \mathcal{T}_k$ and $\mathcal{S}'_k \subset \mathcal{S}_k$, where $T_j \notin \mathcal{T}'_k$ and $S_i \notin \mathcal{S}'_k$. The signals received at node S_i can be expressed similarly.

Data unit d_i sent from S_i to T_j, can be recovered by adding the signals received on $\mathbf{T}_k$ and $\mathbf{R}_k$ as expressed by equations (2) and (3), in addition to $\alpha_j^k u_j$, where u_j is a data unit generated earlier by node T_j, and is stored in its buffer, viz.,

$$\left(\alpha_i^k d_i^t + \sum_{l, S_k \in \mathcal{S}'_k} \alpha_l^k d_l^t + \sum_{l, T_l \in \mathcal{T}'_k} \alpha_l^k u_l^t \right) + \alpha_j^k u_j + \left(\alpha_j^k u_j^r + \sum_{l, S_k \in \mathcal{S}'_k} \alpha_l^k d_l^r + \sum_{l, T_l \in \mathcal{T}'_k} \alpha_l^k u_l^r \right) = \alpha_i^k d_i \tag{4}$$

With all nodes knowing $\vec{\alpha}^k$, data unit d_i can then be easily recovered from the above result. Therefore, if only one primary path between S_i and T_j where $S_i \in \mathcal{S}_k$ and $T_j \in \mathcal{T}_k$ fails, d_i can be recovered by T_j using the above procedure. Data unit u_j can also be recovered by S_i using a similar procedure.

D. Data Recovery Algorithm:

Consider a node with connections that are protected against m link failures using m cycles. Let this set be $\mathbb{C}$.

Case I: *Normal operation (no failures)*

Data is received on the working paths. m copies of the same data can be also recovered from the m cycles in $\mathbb{C}$, hence allowing recovery from transmission errors, which is an added functionality.

Case II: *Recovery from multiple failures on working paths:* Suppose there is a total of n failures in the network, and all of the failures are on working paths. Assume also that m, where $m \leq n$, of the failed working paths are protected by cycle $\mathbf{C}_k$. Also, let $\mathcal{S}_k$ and $\mathcal{T}_k$ denote the set of nodes in $\mathcal{S}$ and $\mathcal{T}$, respectively, using cycle $\mathbf{C}_k$ for protection. Denote by $\overline{\mathcal{S}}_k$ and $\overline{\mathcal{T}}_k$ the nodes in $\mathcal{S}_k$ and $\mathcal{T}_k$, respectively, with failed working paths, i.e., $|\overline{\mathcal{S}}_k| = |\overline{\mathcal{T}}_k| = m$. We consider the data recovery at node $T_i \in \overline{\mathcal{T}}_k$.

Let node T_i be protected by l cycles, where $m \leq l$, and of which, as assumed above, is cycle $\mathbf{C}_k$. Using the approach described above, node T_i adds incoming signals on $\mathbf{T}_k$ and $\mathbf{R}_k$, as well as $\alpha_i^k u_i$ where u_i is a signal that was transmitted in an earlier cycle (see equation (4)) to obtain the following equation

$$\sum_{j, S_j \in \overline{\mathcal{S}}_k} \alpha_j^k d_j \quad . \tag{5}$$

S_j in the above equation is the source of the d_j data unit transmitted on one of the failed paths. Since node T_i is protected by l cycles, it obtains l equations of the form shown in equation (5), and each equation should contain a number of variables (d data units) which correspond to failed connections which are protected by the cycle generating this equation. Therefore, if the total number of d variables, denoted by D, in such equations does not exceed l, and we have at least D linearly independent equations, then d_i can be recovered.

Case III: *Recovery from m failures on working paths, and n failures on protection cycles:*

Note that the failed protection cycles cannot be used to recover from failures. Hence, the m failures can be recovered from if they are protected by $l \geq m$ cycles, which have not failed. The discussion in Case II above still holds.

Referring to the example in Figure 1, in Table II we show three sets of failures and the equations which can be obtained at some of the affected nodes in $\mathcal{T}$. In Table II.(a), the two paths used by connections (S_2, T_2) and (S_3, T_3) fail. Since both connections can withstand at least 2 failures, node T_3 obtains a set of two equations. Node T_2, however, obtains 3 equations, since it is also protected by cycle $\mathbf{C}_3$. Therefore, all nodes involved can recover their affected data units if the equations can be uniquely solved. In Table II.(b), three paths fail, which are used by connections (S_1, T_1), (S_2, T_2), and (S_7, T_7). Only node T_2 is protected against 3 failures, and obtains 3 equations in 3 unknowns, which it can use to recover data unit d_2. However, node T_1 is only able to obtain 2 equations, but in 2 unknowns, since one of the failures (between S_7 and T_7) does not share a cycle with it. Therefore, node T_1 can recover d_1. On the other hand, node T_7 obtains one equation in two unknowns, and it cannot recover d_7. The third example, shown in Table II.(c), is different, since in this case cycle $\mathbf{C}_1$ fails in addition to working paths for connections (S_1, T_1) and (S_2, T_2). Nodes T_1 and T_2 are unable to obtain equations using cycle $\mathbf{C}_1$, and therefore node T_1 obtains one equation only in d_1 and d_2, which are not sufficient to recover d_1. However, node T_2, which is protected using three cycles, obtains two equations in d_1 and d_2 on cycles $\mathbf{C}_2$ and $\mathbf{C}_3$, from which it can recover d_2.

TABLE II

AN EXAMPLE OF FAILURES AND THE SETS OF EQUATIONS OBTAINED AT THE NODES IN $\mathcal{T}$; f_i^j IS THE jTH EQUATION OBTAINED BY NODE T_i.

Failures	T_2	T_3
(S_2, T_2), (S_3, T_3)	$f_2^1(d_2, d_3)$ on $\mathbf{C}_1$ $f_2^2(d_2, d_3)$ on $\mathbf{C}_2$ $f_2^3(d3)$ on $\mathbf{C}_3$	$f_3^1(d_2, d_3)$ on $\mathbf{C}_1$ $f_3^2(d_2, d_3)$ on $\mathbf{C}_2$

(a)

Failures	T_1	T_2	T_7
(S_1, T_1), (S_2, T_2), (S_7, T_7)	$f_1^1(d_1, d_2)$ on $\mathbf{C}_1$ $f_1^2(d_1, d_2)$ on $\mathbf{C}_3$	$f_2^1(d_1, d_2)$ on $\mathbf{C}_1$ $f_2^2(d_1, d_2, d_7)$ on $\mathbf{C}_2$ $f_2^3(d_1, d_2)$ on $\mathbf{C}_3$	$f_7^1(d_2, d_7)$ on $\mathbf{C}_2$

(b)

Failures	T_1	T_2
$\mathbf{C}_1$, (S_1, T_1), (S_2, T_2)	$f_1^1(d_1, d_2)$ on $\mathbf{C}_3$	$f_2^1(d_2)$ on $\mathbf{C}_2$ $f_2^2(d_1, d_2)$ on $\mathbf{C}_3$

(c)

E. Conditions for Data Recovery:

We represent the failed connections, and the protecting cycles using a bipartite graph, $G(V, E)$, where the set of vertices $V = \mathbb{N} \cup \mathbb{C}$, and the set of edges $E \subseteq \mathbb{N} \times \mathbb{C}$. $\mathbb{N}$ is the set of connections to be protected, and $\mathbb{C}$ is the set of protecting cycles. That is, there is an edge from connection $N_i \in \mathbb{N}$ to cycle $\mathbf{C}_k \in \mathbb{C}$ if cycle $\mathbf{C}_k$ protects connection N_i.

Define $\mathbb{C}(N_i)$ as the set of cycles protecting connection N_i, i.e., $\mathbb{C}(N_i) = \{\mathbf{C}_k \in \mathbb{C} : \text{edge } (N_i, \mathbf{C}_k) \in E\}$. Also define $\mathbb{N}(\mathbf{C}_k)$ as the set of connections protected by cycle $\mathbf{C}_k$. That is, $\mathbb{N}(\mathbf{C}_k) = \{N_i \in \mathbb{N} : \text{edge } (N_i, \mathbf{C}_k) \in E\}$. Then, the following four conditions must apply for connection N_i to be protected against m failures, i.e., the end nodes of N_i can recover their data when there are m failures:

C1: Connection N_i must be protected by at least m cycles, i.e.,

$$|\mathbb{C}(N_i)| \geq m \tag{6}$$

This is a necessary, but not a sufficient condition.

C2: The number of connections protected by a cycle $\mathbf{C}_k \in \mathbb{C}(N_i)$ must not exceed the number of cycles protecting connection N_i. That is,

$$|\mathbb{N}(\mathbf{C}_k)| \leq |\mathbb{C}(N_i)| \quad \forall \mathbf{C}_k \in \mathbb{C}(N_i) \tag{7}$$

This condition guarantees that the number of variables in each of the equations obtained on one cycle does not exceed the number of cycles protecting connection N_i. Also, this condition is necessary, but not sufficient.

C3: The number of connections protected by all cycles protecting N_i must not exceed the number of such cycles. That is,

$$|\bigcup_{\mathbf{C}_k \in \mathbb{C}(N_i)} \mathbb{N}(\mathbf{C}_k)| \leq |\mathbb{C}(N_i)| \quad (8)$$

This is to guarantee that the number of *distinct* variables in the set of equations does not exceed the number of equations. This is a necessary condition.
Notice that since $\mathbb{N}(\mathbf{C}_k) \subseteq \bigcup_{\mathbf{C}_k \in \mathbb{C}(N_i)} \mathbb{N}(\mathbf{C}_k)$, condition C3 always implies condition C2.

C4: Any subspace of the vector space consisting of the vectors $\vec{\alpha}^k$, where $k = i_1, i_2, \ldots, i_n$, must correspond to set of linearly independent vectors. This condition guarantees that any two equations of the same set of variables obtained on two different cycles are independent.

The satisfaction of these necessary conditions altogether is a sufficient condition for the protection against m failures, as stated in the theorem below, whose proof is straightforward:

Theorem 1: Connection N_i will be guaranteed protection against any m link failures in the network if and only if conditions C1, C2, C3 and C4 are satisfied.

In Figure 3 we show an example that applies to the network in Figure 1. Figure 3.(a) shows the bipartite graph for the entire network, while Figures 3.(b), 3.(c) and 3.(d) show the graph corresponding to the failures in Table II.(a), II.(b) and II.(c), respectively. Let all vectors $\vec{\alpha}^k$ be chosen to satisfy condition C4. From Figure 3.(b) it can be verified that the four conditions stated above are satisfied to detect the failures of connections (S_2, T_2) and (S_3, T_3). From Figure 3.(c) the conditions for detecting the failures in connections (S_1, T_1) and (S_2, T_2) are satisfied. However, for connection (S_7, T_7), condition C2, and consequently condition C3, are not satisfied. In Figure 3.(d), cycle $\mathbf{C}_1$ does not exist, and T_1 is protected only by cycle $\mathbf{C}_3$. Therefore, condition C2 for connection (T_1, S_1) is violated. T_2 can still recover its d_2 data unit, since all conditions are satisfied. Figure 3.(a), together with conditions C1, C2 and C3, are useful in finding, for each connection N_i, which other connections can fail, and still protect connection N_i. For example, taking connection (S_1, T_1), since it is protected by two cycles, $\mathbf{C}_1$ and $\mathbf{C}_2$, by applying condition C1 we find that if this connection fails it can still tolerate the failure of another connection. Applying condition C2 we find that only one of the connections in the set $\{(S_2, T_2), (S_3, T_3), (S_4, T_4), (S_5, T_5)\}$ and one of the connections in the set $\{(S_2, T_2), (S_6, T_6)\}$ can fail. Applying condition C3 means that only connection (S_2, T_2) can fail with connection (S_1, T_1) and still recover the data of connection (S_1, T_1).

III. Data Recovery Algorithm Under Failure of Two Primary Paths

In this section we show the details of providing protection and data recovery when two primary paths fail, but are protected against two failures.

Consider two protection cycles, $\mathbf{C}_1$ and $\mathbf{C}_2$, over which data is encoded using the two vectors $\vec{\alpha}^1$ and $\vec{\alpha}^2$, respectively. Let

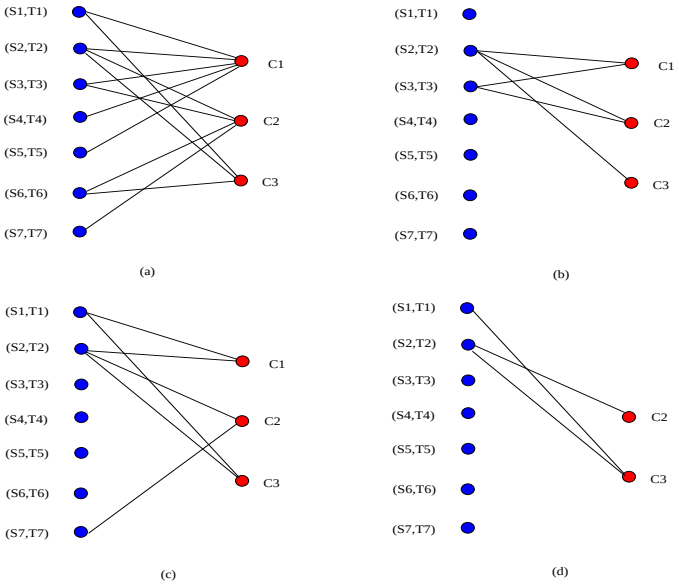

Fig. 3. Using the bipartite graph to verify protection against failures

us first consider the recovery of d_i and d_j data units by nodes $T_{i'}$ and $T_{j'}$, respectively, due to the failure of their working paths to nodes S_i and S_j, respectively. Since other nodes in $\mathcal{T}$, e.g., $T_{k'}$, can remove their received data units, d_k, then nodes $T_{i'}$ and $T_{j'}$ will each calculate the following two combinations from the $\mathbf{C}_1$ and $\mathbf{C}_2$ cycles, respectively.

$$\alpha_i^1 d_i + \alpha_j^1 d_j \quad (9)$$
$$\alpha_i^2 d_i + \alpha_j^2 d_j \quad (10)$$

Without loss of generality, we assume that $i < j$. Note that both of the nodes, $T_{i'}$ and $T_{j'}$, are aware that their working paths have failed, but they do not know which other working paths have failed. Therefore, as far as node $T_{i'}$ is concerned, to solve the above two equations and recover the d_i data units, the following three conditions must be satisfied:

1) Equations (9) and (10) must be linearly independent, which follows from condition C4 in Section II-E.
2) The values of d_i and d_j can be recovered from equations (9) and (10) by using XOR operations only. A sufficient condition for this is that the least significant 1's in the binary representations of each of the two products $\alpha_i^1 \alpha_j^2$ and $\alpha_i^2 \alpha_j^1$ should not occur in the same bit position, for all $i \neq j$.
3) In addition to knowing its own α_i^1 and α_i^2, node $T_{i'}$ must also know α_j^1 and α_j^2.

Condition 3 above can be achieved by using two header fields, $\mathcal{D}$ and $\mathcal{U}$, with each packet, which are bit maps of which d_i and u_i packets are covered by the encoded packet, respectively (see Figure 4). The bits corresponding to the d_i and u_i bits will be set by the nodes in $\mathcal{S}$ and $\mathcal{T}$, which add them, respectively, and will be reset by the nodes in $\mathcal{T}$ and $\mathcal{S}$ which remove them.

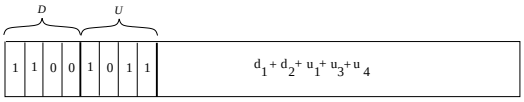

Fig. 4. Data unit map header: data units d_1, d_2, u_1, u_3 and u_4 are XORed.

Conditions 1 and 2 above can be satisfied by using:

$$\begin{aligned} \alpha_i^1 &= 1 \text{ for } i = 1, 2, \ldots, N \\ \alpha_i^2 &= 2^{i-1} \text{ for } i = 1, 2, \ldots, N \end{aligned}$$

This assignment makes equations (9) and (10) independent, and also makes solution of these equations equal to the data unit XORed with a shifted version of itself. That is, multiplying equation (9) by 2^{j-1} and XORing it with equation (10) yields:

$$\delta_i = 2^{i-1}d_i + 2^{j-1}d_i = 2^{i-1}(1 + 2^{j-i})d_i \ , \qquad (11)$$

while multiplying equation (9) by 2^{i-1} and XORing it with equation (10) yields:

$$\delta_j = 2^{i-1}d_j + 2^{j-1}d_j = 2^{i-1}(1 + 2^{j-i})d_j. \qquad (12)$$

These two expressions are used below to solve for d_i and d_j.

Note that the above assignment of α_i^2 is not necessarily an optimal assignment in terms of the packet length (or equivalently, the number of elements in the field of encoded packet symbols), since it requires packet lengths to be equal to s, the packet length, plus $N-1$ bits. This is in addition to the use of $2N$ bits for the data unit maps described above. Using other carefully selected assignments can reduce the number of additional bits to about $\log_2 N + 2N$, at the expense of increased processing in order to recover the data units. Notice that in the case of $\alpha_i^2 = 2^{i-1}$, multiplying by α_i^2 is a simple shift operation. Moreover, as shown below, to recover d_i, or d_j, from equations (11) and (12), respectively, a maximum of N XOR operations are needed, which is less than the $N \log_2 N$ operations needed with other more optimal assignments.

A. Data Recovery Algorithm

With the assumption that $i < j$, to recover d_i, which is M bits long, node $T_{i'}$ applies the procedure in Algorithm 1 to the data unit δ_i in equation (11). In this procedure, node $T_{i'}$ sets $d_i(0)$ to $\delta_i(i-1)$. Then, it multiplies $d_i(0)$ by 2^{j-1}, and adds the product to $\delta_i(j-1)$. It repeatedly applies this procedure to all remaining bits in δ_i, which will finally yield d_i. Similarly, node $T_{j'}$ applies the same procedure to the δ_j data unit in equation (12) in order to recover d_j.

We show an example in Figure 5, in which $M = 4$, $d_1 = 1101$ and $d_3 = 0110$. In the application of the algorithm, the dashed lines indicate how the XOR operations are applied.

IV. Conclusions

This paper has extended the approach presented in [1] for 1+N protection in order to protect against multiple link failures, and presented necessary and sufficient conditions for such protection. A procedure for data recovery when the network is protected against two link failures was presented. The strategy can be applied to provide protection under a number of scenarios including protection against multiple failures, shared risk link group, and non-disjoint primary paths.

Algorithm 1: Algorithm to recover data unit d_i from $\delta_i = 2^{i-1}(1 + 2^{j-i})d_i$

Input: δ_i, i, and j
Output: $d_i = \{d_i(0), d_i(1), \ldots, d_i(M-1)\}$
for $(k = 0; k < M; k++)$ **do**
  $d_i(k) = \delta_i(k + i - 1)$;
  $\delta_i(k + j - 1) = \delta_i(k + j - 1) \oplus d_i(k)$;

$$\begin{aligned} d_1 &= 1101 \\ d_3 &= 0110 \\ d_1 + d_3 &= 1011 \\ d_1 + 2^2 d_3 &= 010101 \\ \delta_1 &= 2^2(d_1 + d_3) + d_1 + 2^2 d_3 \\ &= d_1 + 2^2 d_1 = 111001 \end{aligned}$$

Algorithm:

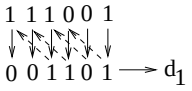

Fig. 5. An example of applying Algorithm 1 with $i = 1$, $j = 3$ and $M = 4$.

References

[1] A. E. Kamal, "1+n protection in optical mesh networks using network coding on p-cycles," in *Proceedings of the Conference on Global Communications (GLOBECOM)*, 2006.

[2] D. Stamatelakis and W. D. Grover, "Ip layer restoration and network planning based on virtual protection cycles," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 10, pp. 1938–1949, 2000.

[3] W. D. Grover, *Mesh-based survivable networks : options and strategies for optical, MPLS, SONET, and ATM Networking*. Upper Saddle River, NJ: Prentice-Hall, 2004.

[4] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, pp. 1204–1216, July 2000.

[5] P. Sebos, J. Yates, G. Hjalmtysson, and A. Greenberg, "Auto-discovery of shared risk link groups," in *Optical Fiber Conference*, 2001.

[6] D. Schupke and R. Prinz, "Performance of path protection and rerouting for wdm networks subject to dual failures," in *Optical Fiber Conference*, pp. 209–210, 2003.

[7] S. Kim and S. Lumetta, "Evaluation of protection reconfiguration for multiple failures in wdm mesh networks," in *Optical Fiber Conference*, pp. 210–211, 2003.

[8] J. Zhang, K. Zhu, and B. Mukherjee, "Backup provisioning to remedy the effect of multiple link failures in wdm mesh networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, pp. 57–67, Aug. 2006.

[9] H. Choi, S. Subramaniam, and H.-A. Choi, "Loopback recovery from double-link failures in optical mesh networks," *IEEE/ACM Transactions on Networking*, vol. 12, pp. 1119–1130, Dec. 2004.

[10] W. He, M. Sridharan, and A. K. Somani, "Capacity optimization for surviving double-link failures in mesh-restorable optical networks," *Photonic Network Communications*, vol. 9, pp. 99–111, Jan. 2005.

[11] Y. Liu, D. Tipper, and P. Siripongwutikorn, "Approximating optimal spare capacity allocation by successive survivable routing," *IEEE/ACM Transactions on Networking*, vol. 13, pp. 198–211, Feb. 2003.