1+N Protection in Mesh Networks Using Network Coding over p-Cycles

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

***Abstract*— p-Cycles have been proposed for pre-provisioned 1:N protection in optical mesh networks. Although the protection circuits are preconfigured, the detection of failures and the rerouting of traffic can be a time consuming operation. Another survivable mode of operation is the 1+1 protection mode, in which a signal is transmitted on two link disjoint circuits, and the destination chooses the strongest of the two signals, hence recovery from failures is instantaneous. The disadvantage is the large number of protection circuits. In this paper we introduce a new concept in protection: 1+N protection, in which a p-Cycle can be used to protect a number of bidirectional connections. Data from different circuits are combined using network coding, which can be implemented in a number of technologies. The maximum outage time under this protection scheme is limited to no more than twice the p-Cycle propagation delay.**

I. Introduction

With the use of optical fibers in network backbones, which are usually configured as arbitrary mesh topologies, large amounts of bandwidth are provided on a single fiber, and huge amounts of traffic are carried on the fiber, especially if wavelength division multiplexing (WDM) is used. The failure of a single fiber can therefore affect a large number of users and connections. It is therefore imperative that when any part of the network fails that the network will continue to operate. This is referred to as *network survivability*.

Techniques for network survivability can be classified as *Predesigned Protection* and *Dynamic Restoration* techniques [1]. In predesigned protection, bandwidth is reserved in advance so that when a failure takes place, backup paths which are pre-provisioned, 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; and 1:N protection, which is similar to 1:1, except that one path is used to protect N paths. A generalization of 1:N is the M:N, where M protection paths are used to protect N working paths. Under dynamic restoration, capacity is not reserved in advance, but when a failure occurs spare capacity is discovered, and is used to reroute the traffic affected by the failure. Protection techniques can recover from failures quickly, but require significant amounts of resources. On the other hand, restoration techniques are more cost efficient, but are much slower than their protection counterparts.

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

Recently, the concept of p-Cycles has been introduced in [2], [3] to provide 1:N protection to connections with the same transport capacity, e.g., DS-3. p-Cycles provide protection against single link failures to a connection with its two end nodes being on the cycle. However, under p-Cycles, and because of the shared protection, failures must still be detected, and traffic must be rerouted on the cycle. This adds complexity to the management plane, and prolongs the failure recovery time.

This paper introduces a strategy for using p-Cycles to provide 1+N protection against single link failures in optical mesh networks. That is, to transmit signals from N connections on one common channel, such that when a failure occurs, the end nodes of the connection affected by the failure will be able to recover the signals affected by the failure. To be able to achieve this, we trade computation for communication. That is, by performing additional computation within the network, in the form of *network coding*, we are able to achieve the desired protection. Hence, to provide survivability, failures need not be detected explicitly, and rerouting of the signal is not needed. Both the management and control planes in this case will be simpler, as they only need to detect the failure for the purpose of repairing it.

Our proposed scheme will provide two copies of the same signal on two disjoint paths. One path is the primary working path. The second path, however, is a virtual path, which is still disjoint from the first primary path. What we mean by a virtual path is a set of paths on which the signal is transmitted with other signals, but there is enough information to recover the target signal from those transmissions. This scheme has the following properties:

1) Protection against a single link failure is guaranteed.
2) p-Cycles which are typically employed for 1:N protection, are used to provide 1+N protection in the sense that a signal can be received on two link disjoint paths, such that if a link fails on one of the paths, the signal can still be received on the other path, where the backup path is multiplexed.
3) Resuming data reception on the protection path is within twice the propagation delay around a p-Cycle.

In this paper we introduce the basic concepts and theoretical bases of the strategy, and how it can be used to provide 1+N protection against single link failures. Due to space

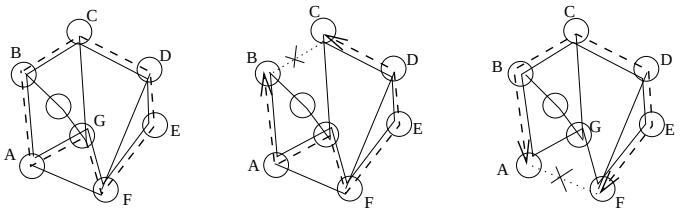

Fig. 1. p-Cycle concept: (a) a cycle (dashed lines) traversing nodes A-G, and protecting circuits (solid lines) on the same physical path as the cycle, and on straddling links; (b) protection of a failure (dotted line) on the cycle; (c) protection of a failure (dotted line) on a straddling path.

restrictions, we leave the details of implementation for a future study.

The rest of the paper is organized as follows. In Section II we provide a brief background to p-Cycles and network coding. In Section III we introduce a few operational assumptions. We illustrate the basic concept of our strategy by giving an example of using network coding to provide protection against a single link failure in Section IV. In Section V we show the general strategy to encoding and decoding data units on p-Cycles in order to provide protection for bidirectional unicast connections. We discuss the issue of timing and synchronization of encoded and decoded data, and we show that the outage time, which is the time between the loss of the direct signal, and the recovery of the same signal on the protection path, is limited to no more than twice the delay on the p-Cycle. Finally, in Section VI we conclude the paper with a few remarks.

II. Background

A. Background on p-Cycles

The p-Cycle concept [2], [3] is similar to the Bidirectional Line-Switched Ring (BLSR), since both of them have a cyclic structure. However, the p-Cycle concept has a higher protection coverage, since the spare capacity reserved on the cycle covers working capacity on the cycle, as well as working capacity on straddling links (see Figure 1). Since the protection capacity can be used to protect multiple connections, the p-Cycle belongs to the 1:N protection. The endpoints of the failure are responsible for detecting the failure, and for rerouting the traffic on the p-Cycle.

There are two types of p-Cycles: *link p-Cycles*, which are used to protect the working capacity of a link, and this is the type shown in Figure 1, and *node-encircling p-Cycles*, which protect paths traversing a certain node against the failure of this node.

p-Cycles are embedded in mesh networks, and several algorithms have been introduced in the literature to select the p-Cycles which consume the minimum amounts of spare capacity, e.g., see Chapter 10 in [3]. p-Cycles are very efficient in protecting against link failures, and the protection capacity reserved by p-Cycles achieves an efficiency that is close to that achievable in mesh-restorable networks. However, the pre-provisioning of spare capacity makes p-Cycles much faster to recover from network element failures. p-Cycles can be used at a number of layers including the Optical layer, the SONET layer, or the IP layer.

In this paper we will use p-Cycles to protect against failures in a 1+N manner, rather an 1:N. That is, we allow two transmissions of the same signals of multiple connections on the p-Cycles in a bandwidth efficient manner, and to allow the receiver to select the better of the two signals. The backup signals are multiplexed using network coding. Our approach can also be used at any layer that transmits encapsulated data units including the SONET layer, especially Next Generation SONET, the ATM layer, and the IP layer.

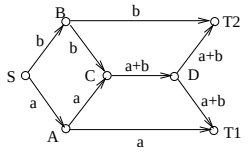

Fig. 2. An example of network coding

B. Background on Network Coding

Network coding refers to performing linear coding operations on traffic carried by the network at intermediate network nodes. In this case, a node receives information from all, or some of its input links, encodes this information, and sends the information to all, or some of its output links. This approach can result in enhancing the network capacity, hence facilitating the service of sessions which cannot be otherwise accommodated. This is especially true when the service mode is multicast, as shown in the example of Figure 2 in which node S transmits to nodes T1 and T2, and each link in the network has a capacity of one data unit per time unit. Data units a and b are delivered to both T1 and T2 by adding a and b at node C, where the addition is modulo 2. Both a and b are recovered at T1 and T2 by adding the explicitly received data units (a and b, respectively), to a+b. The network can then achieve a capacity of two data units per time unit.

The concept of network coding for multicast sessions was introduced in the seminal paper by Ahlswede et al. [4]. The problem of network coding was formulated as a network flow problem in [5] and a link cost function was included in the formulation in [6]. Reference [7] introduced an algebraic characterization of linear coding schemes that results in a network capacity that is the same as the max-flow min-cut bound, when multicast service is used. The authors show that failures can be tolerated through a static network coding scheme under multicasting, provided that the failures do not reduce the network capacity below a target rate. Reference [8] includes an introduction to network coding principles.

Our objective in this paper is to use network coding with a group of unicast sessions in order to provide protection for such connections.

III. Operational Assumptions

In this section we introduce a number of operational assumptions.

- A p-Cycle protecting a number of connections passes through all end nodes of such connections, and it protects connections with the same transport capacity unit, e.g., DS-3. Therefore, the p-Cycle links themselves have the same transport capacity.
- The p-Cycle is terminated, processed, and retransmitted at each node on the cycle.
- We assume that all connections are bidirectional.
- It is assumed that data units are fixed in size[1].
- The scheme presented in this paper is designed to protect against single link failures.
- When a link carrying active circuits fails, the tail node of the link will receive empty data units.

This paper presents the concepts of using network coding on p-Cycles to achieve 1+N protection. The paper does not address implementation issues, which are the subject of a future study. However, it should be pointed out that this strategy can be implemented using the GFP [9] protocols of NGS. The strategy can also be implemented using IP.

It should be pointed out that all addition operations (+) in this paper are **modulo two**, i.e., XOR operations.

IV. An Illustrative Example

In this section we illustrate our basic idea using a simple example. As stated above, our objective is to provide each destination with two signals on two link disjoint paths, such that the network can withstand any single link failure. For the sake of exposition, we first consider unidirectional connections, and then extend it to bidirectional ones.

The example is shown in Figure 3.(a), and there are three unidirectional connections from source S_i to destination T_i, for $i = 1, 2, 3$. To simplify the example, we assume that all sources and their corresponding destinations are ordered from left to right. Assume that each connection requires one unit of capacity. Let us also assume that data units d_1, d_2 and d_3 are sent on those connections. A p-Cycle is preconfigured that includes all three sources and destinations, as shown in the figure. Data units d_i will be transmitted three times: once on the primary working path, and twice and in opposite directions on the p-Cycle. One of the transmissions on the p-Cycle is by the original transmitter of the data unit, S_i, and the other is by the receiver, T_i. We distinguish between those last two data units by referring to them as *transmitted* and *received* d_i units, viz., d_i^t and d_i^r, respectively.

On the p-Cycle, the following takes place:

1) Node S_1 transmits d_1^t in the clockwise direction. Node S_2 will add its own data unit, d_2^t to d_1^t which it receives on the p-Cycle, where the addition is modulo 2, and transmits $d_1^t + d_2^t$ on the p-Cycle, also in the clockwise direction. Node S_3 will repeat the same operation, and will add d_3^t to $d_1^t + d_2^t$, and transmits the sum on the p-Cycle. That is, node T_3 receives $d_1^t + d_2^t + d_3^t$ on the p-Cycle, and in the clockwise direction.

[1]If data units are not fixed in size, they can be accommodated by adding data units based on the data unit with the largest size. Shorter data units are extended by adding trailing, or leading zeroes.

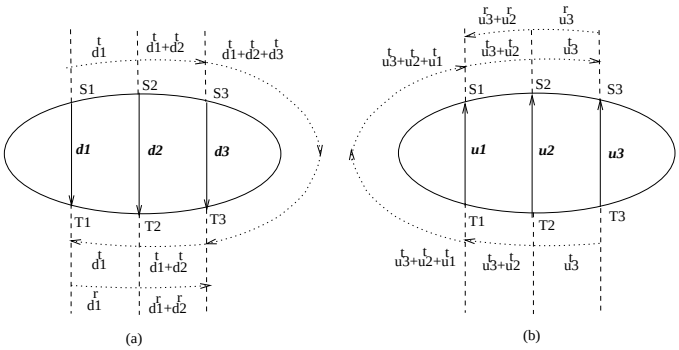

Fig. 3. An example of the use of network coding on p-Cycles to protect against single link failures: (a) the sources are at S_i, and the destinations are at T_i nodes; (b) the source are at T_i, and the destinations are at S_i nodes.

2) On the same direction of the p-Cycle, but at the destinations, when destination T_3 receives $d_1^t + d_2^t + d_3^t$, and receives d_3 on the working path, it adds d_3 to $d_1^t + d_2^t + d_3^t$ to obtain $d_1^t + d_2^t$, and forwards it to T_2. Node T_2 will also add d_2, which it receives on the working path, to $d_1^t + d_2^t$ to recover d_1^t, which it transmits on the same p-Cycle to T_1. T_1 removes d_1^t from the clockwise cycle.
3) Also, when node T_1 receives d_1 on the working path, it sends it on the p-Cycle, but in the counter-clockwise direction. It will be referred to as d_1^r. Node T_2, when it receives d_2 on the working path, it adds it to d_1^r, and transmits $d_1^r + d_2^r$ on the p-Cycle, also in the counter-clockwise, direction.

Based on the above, it is obvious that in the absence of failures, each destination node, T_i, for $i = 1, 2, 3$, receives two copies of d_i:

1) One copy on the working path, and
2) The second copy is obtained by adding $\sum_{j=1}^{i} d_j^t$ which it receives on the clockwise p-Cycle to $\sum_{j=1}^{i-1} d_j^r$, which is receives on the counter-clockwise cycle. This is what we refer to a *virtual copy* of d_i.

In this case, timing considerations have to be taken into account, as will be discussed in the next section.

When a failure occurs, it will affect at most one working path, e.g., working path i. In this case, we assume that T_i will receive an empty data unit on the working path. Therefore, T_i will be able to recover d_i by using the second virtual copy described above, i.e., by adding $\sum_{j=1}^{i} d_j^t$ and $\sum_{j=1}^{i-1} d_j^r$. A failure on the p-Cycle will not disrupt communication on primary working paths.

The case in which information is sent in the opposite direction, i.e., from T_i to S_i is shown in Figure 3.(b). Data units in this case are labeled u_i, and similar to d_i data units, u_i^t and u_i^r distinguish between newly transmitted and received u_i data units.

We refer to a bidirectional p-Cycle as a *full* cycle, and a one directional cycle is a *half* p-Cycle. In each of the above two examples, less than a full p-Cycle is used. In order to support bidirectional communication, the two approaches above have to be combined. In this case, less than three half p-Cycles, or

1.5 full p-Cycles are used. That is, one half p-Cycle (the outer one) is shared by both d_i^r and u_i^r data units. However, this can be accomplished because of the ordering of S_i and T_i that we enforced in this example. In the general case, combining the two bidirectional sessions would require two full p-Cycles. However, by combining u_i and d_j signals on the same link, it is possible to reduce the number of p-Cycles to one full cycle, hence the name 1+N protection. This will be illustrated next.

V. Network Coding Strategy on p-Cycles

In this section we introduce our general strategy for achieving 1+N protection in optical mesh networks.

A. The Strategy

In the examples shown in the previous section, we presented a special case in which the working connections were ordered from left to right. However, in this section we introduce a strategy for general connections. We assume that there are N bidirectional unicast connections, where connection i is between nodes A_i and B_i. We define the sets $\mathbb{A} = \{A_i | 1 \leq i \leq N\}$ and $\mathbb{B} = \{B_i | 1 \leq i \leq N\}$[2]. We denote the data unit transmitted from A_i to B_i as d units, and the data unit transmitted from B_i to A_i as u units.

Before describing the procedure, it should be pointed out that the basic principles for receiving a second copy of data u_i by node A_i, for example, is to receive on two opposite directions the signals given by the following two equations:

$$\sum_{j,\ A_j \in \mathbb{A}'} u_j \qquad \text{and} \qquad u_i + \sum_{j,\ A_j \in \mathbb{A}'} u_j$$

for some $\mathbb{A}' \subset \mathbb{A}$, $A_i \notin \mathbb{A}'$ and the sum is modulo 2. In this case, A_i can recover u_i by modulo 2 addition of the above two equations.

Our procedure goes through the following steps:

A.1 p-Cycle Construction and Node Assignment to Cycles:

1) Find a full p-Cycle. The full p-Cycle consists of two unidirectional half p-Cycles in opposite directions (more on this in item 3 below)[3]. These two p-Cycles do not have to traverse the same links, but must traverse the nodes in the same order.
2) Construct two sequences of nodes, $\mathcal{S} = (S_1, S_2, \ldots, S_N)$ and $\mathcal{T} = (T_1, T_2, \ldots, T_N)$ of equal lengths, N. All elements of $\mathcal{S}$ and $\mathcal{T}$ are in $\mathbb{C} = \mathbb{A} \cup \mathbb{B}$, such that if two nodes communicate, then they must be in different sequences. We use the simple procedure shown in Algorithm 1 to construct the sequences.

   We arbitrarily select the sequence of nodes in $\mathcal{S}$ to be in the clockwise direction, and the sequence of nodes in $\mathcal{T}$ to be in the counter-clockwise direction. We also start with any node in C as S_1, and we label this node as A_1. All nodes in $\mathcal{S}$ belong to the set $\mathbb{A}$, and all nodes in $\mathcal{T}$ belong to the set $\mathbb{B}$. Node T_1 will always be the one to the left of node S_1. Figure 4 shows how ten nodes, in five connections are assigned to $\mathcal{S}$ and $\mathcal{T}$.

   A node S_i in $\mathcal{S}$ (T_i in $\mathcal{T}$) transmits d_i (u_i) data units to a node in $\mathcal{T}$ ($\mathcal{S}$).
3) The two half p-Cycles are a clockwise half p-Cycle, and a counter-clockwise half p-Cycle, which are used as follows:
   a) A half p-Cycle in the clockwise direction, **T**. On this half cycle d_i units newly generated by nodes in $\mathcal{S}$, and u_i units newly generated by nodes in $\mathcal{T}$ are encoded and transmitted as d_i^t and u_i^t, respectively. The d_i^t and u_i^t data units are decoded and removed by the corresponding receivers in $\mathcal{T}$ and $\mathcal{S}$, respectively.
   b) A half p-Cycle in the counter-clockwise direction, **R**. On this half cycle, d_i units received on the primary working paths by nodes in $\mathcal{T}$, and u_i data units received, also on the primary working paths, by nodes in $\mathcal{S}$ are encoded and transmitted as d_i^r and u_i^r, respectively. The d_i^r and u_i^r data units are decoded and removed by the corresponding transmitters in $\mathcal{S}$ and $\mathcal{T}$, respectively.

   Note that the encoding and decoding operations referred to above are simple modulo 2 addition operations of data units to be transmitted and the signals received on such cycles, as will be explained below.

Transmissions occur in rounds, such that d_i^t data units which are encoded together and transmitted on the p-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 u_i^t data units are transmitted, as well as rounds for transmitting sums of d_i^r, and sums of u_i^r data units.

Algorithm 1: Algorithm for constructing the sequences $\mathcal{S}$ and $\mathcal{T}$

Initialization:
- $\mathcal{S} = \mathcal{T} = (\ )$; //initialize empty sequences
- $i = 1,\ j = N$;
- $\mathbb{C} = \mathbb{A} \cup \mathbb{B}$;
- $S_1 = A_1$;
- // select first node in $\mathcal{S}$, and traverse p-Cycles
- $i = i + 1$;
- $\mathbb{C} = \mathbb{C} - \{A_1\}$;

while $\mathbb{C} \neq \phi$ **do**
- c = next node on p-Cycles in clockwise direction;
- **if** *c communicates with a node in $\mathcal{S}$* **then**
  - $T_j = c$;
  - $j = j - 1$;
- **else**
  - $S_i = c$;
  - $i = i + 1$;
- $\mathbb{C} = \mathbb{C} - \{c\}$;

[2]The choice of A_i and B_i is arbitrary, as long as A_i and B_i communicate.

[3]We assume that such p-Cycles exist, but if they do not exist, we find the largest subset of connections for which such p-Cycles exist.

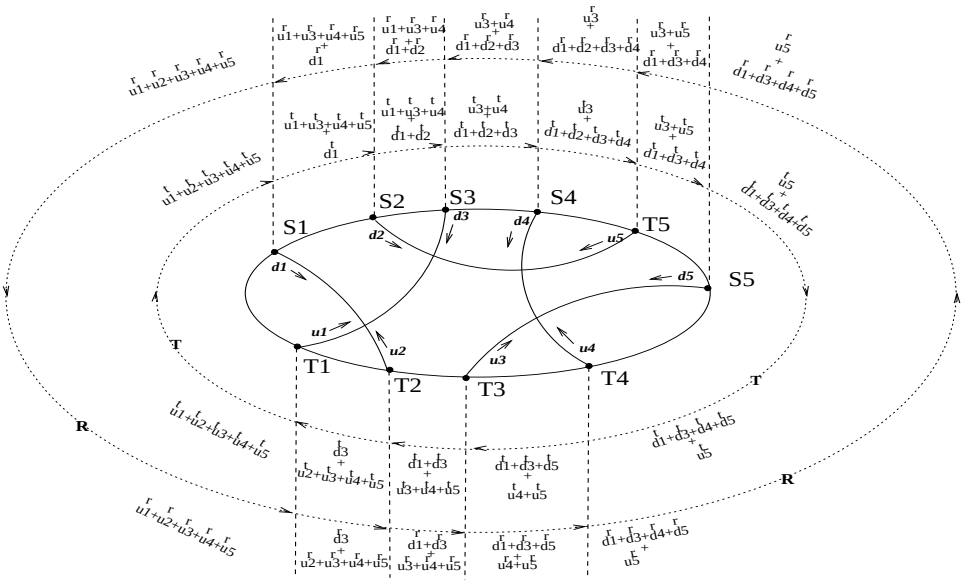

Fig. 4. An example of the application of the network coding procedure to a p-Cycle.

A.2 Encoding Operations:

The network encoding operation is executed by the nodes in $\mathcal{S}$ and $\mathcal{T}$ as follows (assuming no link failures):

1) Node S_i:
   a) The node will add the following data units to the signal received on **T**:
      - Data unit d_i^t, which is newly generated by S_i.
      - Data unit u_j^t, which is received on the primary path from T_j.

      The result is transmitted on the outgoing link in **T**.
   b) The node will add the following data units to the signal received on **R**, and will transmit the result on the outgoing link in **R**.
      - Data unit d_i^r, which it transmitted in an earlier round.
      - Data unit u_j^r, which it received on the primary path from T_j.
2) Node T_i will perform similar operations:
   a) The node will add the following data units to the signal received on **T**:
      - Data unit u_i^t, which is newly generated by T_i.
      - Data unit d_j^t, which is received on the primary path from S_j.

      The result is transmitted on the outgoing link in **T**.
   b) The node will add the following data units to the signal received on **R**:
      - Data unit u_i^r, which it transmitted in an earlier round.
      - Data unit d_j^r, which it received on the primary path from T_j.

      Also, the result is transmitted on the outgoing link in **R**.

To understand the encoding and decoding operations, we first define the following:

- $T(S_i)$: node in $\mathcal{T}$ transmitting and receiving from S_i.
- $S(T_i)$: node in $\mathcal{S}$ transmitting and receiving from T_i.
- $D(Tx)_i$ = sum of d data units transmitted by $S_1,\ S_2, \ldots, S_i$ on half cycle **T** which have not yet been removed by nodes $T(S_1), T(S_2), \ldots, T(S_i)$.
- $U(Tx)_i$ = sum of u data units transmitted by $T_i,\ T_{i+1}, \ldots, T_N$ on half cycle **T** which have not yet been removed by nodes $S(T_i), S(T_{i+1}), \ldots, S(T_N)$.
- $U(Rx)_i$ = sum of u data units received by $S_i,\ S_{i+1}, \ldots, S_N$ and transmitted on half cycle **R** which have not yet been removed by nodes $T(S_i), T(S_{i+1}), \ldots, T(S_N)$.
- $D(Rx)_i$ = sum of d data units received by $T_1,\ T_2, \ldots, T_i$ and transmitted on half cycle **R** which have not yet been removed by nodes $S(T_1), S(T_2), \ldots, S(T_i)$.

It should be noted that all data units in each of the above sums have the same sequence number, as explained above.

Now, the above procedure can be explained as follows, with the help of the example in Figure 4:

1) In step 1a above, node S_i receives $D(Tx)_{i-1} + U(Tx)_j$ on the incoming link on **T**. Node T_j is the node in $\mathcal{T}$ next to S_i in the counter-clockwise direction. For example, for S_2 in Figure 4, it is T_1, and for S_5, it is T_5. The addition operations will add d_i to $D(Tx)_{i-1}$, and will remove $u_{T(S_i)}$ from $U(Tx)_j$. This will result in $D(Tx)_i + U(Tx)_j$ at the output of node S_i, which will be transmitted on the outgoing link on **T**.
   Node S_3 in Figure 4 adds d_3, which is transmitted on the outgoing link. However, adding u_1, where $T(S_3) = T_1$, removes it and is therefore not transmitted on **T**.
2) Also, in step 1b, node S_i receives $U(Rx)_{i+1} + D(Rx)_j$ on the incoming link on **R**. Node T_j is the node in

$\mathcal{T}$ which is next to S_i in the clockwise direction. For example, in Figure 4, for S_3 it is T_5, and for S_5, it is T_4. After the addition operation, $u_{T(S_i)}$ is added, and d_i is removed. The node outputs $U(Rx)_i + D(Rx)_j$ on **R**. In Figure 4, at node S_3, the addition of d_3 to the incoming signal on **R** removes it, while the addition of u_1, where $T_1 = T(S_3)$ adds it to the signal which is transmitted on the outgoing link on **R**.

3) In step 2a, node T_i receives $U(Tx)_{i+1} + D(Tx)_j$ on the incoming link of **T**, where node S_j is the node in $\mathcal{S}$ next to T_i in the counter-clockwise direction. For example, in Figure 4, for T_3 it is node S_5. The addition operation adds u_i, and removes d_j, where $S_j = S(T_i)$, and produces $U(Tx)_i + D(Tx)_j$, which is transmitted on the outgoing link of **T**.
   In Figure 4, T_2 adds u_2, and removes d_1
4) Finally, in step 2b, node T_i receives $D(Rx)_{i-1}+U(Rx)_j$ on the incoming link of **R**, where S_j is the node in $\mathcal{S}$ next to T_i in the clockwise direction. For example, for T_5, it is S_5, and for T_3, it is S_1.
   The addition operation adds d_j, and removes u_i, where $S_j = S(T_i)$. The result is $D(Rx)_i + U(Rx)_j$, which is transmitted on the outgoing link of **R**.
   In Figure 4, T_3 adds d_5, and removes u_3.

A.3 Recovery from Failures:

The strategy presented in this paper recovers from a single link failure on any of the N primary paths. Suppose that a link on the path between nodes S_i and T_j fails. In this case, S_i does not receive u_j on the primary path. However, it can recover u_j by adding

- $D(Tx)_{i-1} + U(Tx)_j$ which is received on **T**,
- $U(Rx)_{i+1} + D(Rx)_j$, that it receives on **R**, and
- d_i that it generated and transmitted earlier.

For example, at node S_3 in Figure 4, adding the signal received on **T** to the signal received on **R**, and d_3, then u_1 can be recovered, which was generated by $T_1 = T(S_3)$.

Similarly, node T_j can recover d_i by adding

- $U(Tx)_{i+1} + D(Tx)_j$ which it receives on **T**,
- $D(Rx)_{i-1} + U(Rx)_j$ which is received on **R**, and
- u_i that it generated and transmitted earlier.

Node T_2 adds the signals on **T** and **R**, and the u_2 it generated earlier to recover d_1. Note that the signals on **T** and **R** which are added together must have the same round number, as explained earlier.

B. Timing Considerations

For the above procedure to work properly, u_i units added and removed at a node should be the same as those carried by the p-Cycle. For this reason, nodes operate in rounds, where in round n, u_i units belonging to this round are added or deleted. The same thing applies to d_i units. Node S_1 can start the first round on **T**, and the remaining nodes $\mathcal{S}$ and $\mathcal{T}$ follow. When data in the first round arrives at node T_1 on **T**, it starts transmitting data received in round 1 on **R**, and all the nodes in $\mathcal{T}$ and $\mathcal{S}$ follow. The start of rounds can be indicated in different ways depending on the protocol.

Since primary paths are usually chosen as the shortest paths. Therefore, data arriving at a destination node over the primary path will do so before data sent over the p-Cycle will arrives. Finally, it should be noted that data must be buffered for no more than two p-Cycle propagation times, which is the maximum outage time, i.e., the delay to receive the backup copy of the data transmitted when the primary working path fails. MSPP devices which can accommodate a 128ms differential delay can support this implementation.

C. Advantages of the Proposed Strategy

The proposed strategy has a number of advantages:

- The strategy provides 1+N protection against single link failures, in which the resources required are on the order of those required for protecting a single connection.
- Receiving nodes would receive the same signal on two different paths, and can therefore select the stronger of the two signals, i.e., without detecting the failure location.
- The management and control planes will be simplified since they do not need to detect the location of the failure, or reroute the signals in order to be able detect and recover from the failure.
- Since signals will be received twice, and on two different paths, this strategy can also be used for error detection and correction.

VI. Conclusions

This paper has presented the principles of a scheme for achieving 1+N protection against single link failures by using network coding on p-Cycles. Data units are coded at sources and destinations, and transmitted in opposite directions on p-Cycles, such that when a link on the primary path fails, data can be recovered from the p-Cycle using simple modulo 2 addition. The strategy allows fast and graceful recovery from failures, and can be implemented at a number of layers and using a number of protocols including ATM, IP, or NGS.

References

[1] D. Zhou and S. Subramaniam, "Survivability in optical networks," *IEEE Network*, vol. 14, pp. 16–23, Nov./Dec. 2000.

[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] T. Ho, D. R. Karger, M. Medard, and R. Koetter, "Network coding from a network flow perspective," in *Intl. Symp. on Info. Theory*, 2003.

[6] D. S. Lun, M. Medard, T. Ho, and R. Koetter, "Network coding with a cost criterion," tech. rep., MIT LIDS Technical Report P-2584, Apr. 2004.

[7] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, pp. 782–795, Oct. 2005.

[8] C. Fragouli, J.-Y. LeBoudec, and J. Widmer, "Network coding: An instant primer," *ACM Computer Communication Review*, vol. 36, pp. 63–68, Jan. 2006.

[9] E. Hernandez-Valencia, M. Scholten, and Z. Zhu, "The generic framing procedure (gfp): An overview," *IEEE Communications*, vol. 40, pp. 63–71, May 2002.